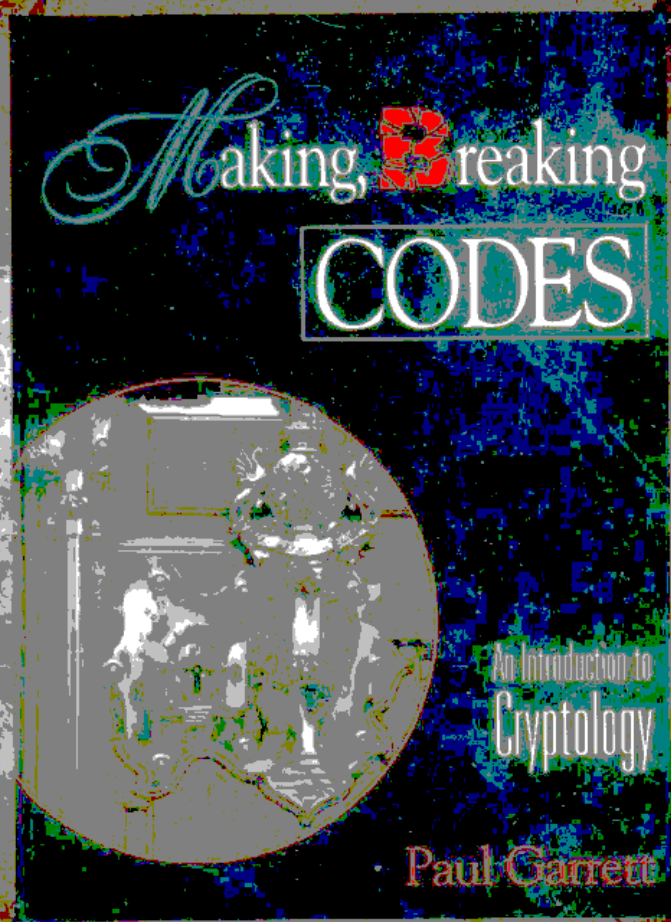
计 算 机 科 学 丛 书

密码学导引

(美) Paul Garrett 明尼苏达大学 著 吴世忠 宋晓龙 郭涛 等译



Making, Breaking Codes
An Introduction to Cryptology

机械工业出版社
China Machine Press

本书着重介绍现代密码学的加密思想及其实现方法，内容涉及数论、概率论、抽象代数，还描述了加密算法的思想及复杂度理论。本书可作为高校信息安全专业密码学导论课程的教材。

本书主要内容包括：

- 介绍密码学的历史沿革，剖析古典的加密算法为何会被现代的加密算法所取代
- 古典和现代密码学体系的数学理论基础
- 各种加密算法的密码分析方法
- 展望密码编码领域的未来

作者简介

Paul Garrett 1973年21岁时获普渡大学硕士学位，1977年于普林斯顿大学获博士学位，之后在耶鲁大学任教。1979～1981年在加州大学伯克利分校获得美国国家科学基金会资助的博士后奖学金，1979年成为斯坦福大学副教授，自1982年起Paul Garrett开始在明尼苏达大学授课，1987年成为该校教授，目前他是该校数学系研究生教学主任，指导着13位博士。Paul Garrett主要的研究方向是数论，并由此而对密码学、计算及算法产生了兴趣。著有《Holomorphic Hilbert Modular Forms》、《Buildings and Classical Groups》，以及两本密码学教材。

Paul Garrett欢迎读者们访问他的本人网址http://www.math.umn.edu/~garrett/，并希望能跟志同道合者深入探讨和研究。

www.PearsonEd.com

ISBN 7-111-12478-2

9 787111 124788

HZ BOOKS 华章图书

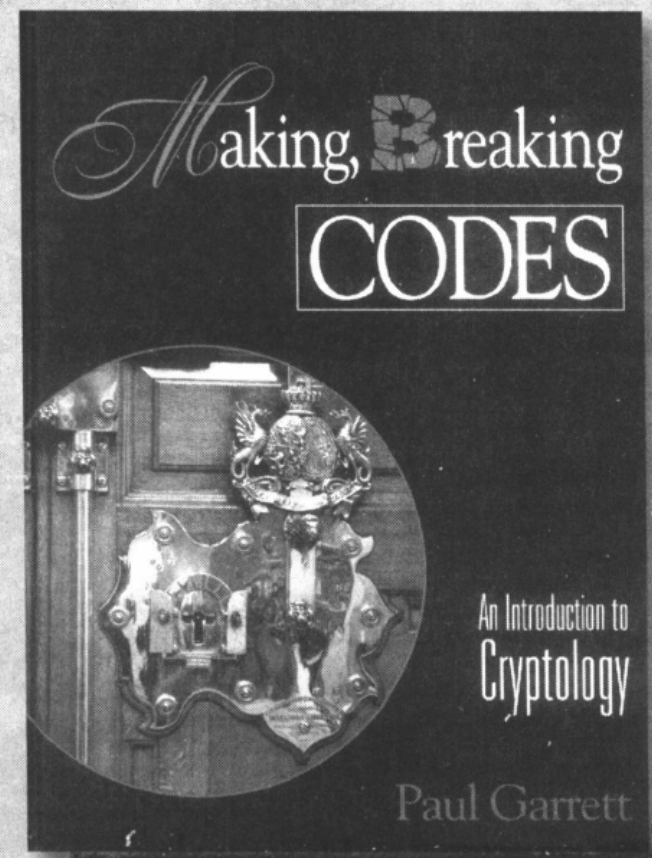
网上购书：www.china-pub.com
北京市西城区百万庄南街1号 100037
读者服务热线：(010)68995259，68995264
读者服务信箱：hzedu@hzbook.com
http://www.hzbook.com

ISBN 7-111-12478-2/TP·2770
定价：39.00元

计 算 机 科 学 丛 书

密码学导引

(美) Paul Garrett 明尼苏达大学 著 吴世忠 郭涛 宋晓龙 等译



Making, Breaking Codes
An Introduction to Cryptology

TN918.2/6
05.10.02

机械工业出版社
China Machine Press

0767643

本书着重介绍现代密码学的加密思想及其实现方法，内容涉及数论、概率论、抽象代数、加密算法的思想及复杂度理论。本书介绍了密码学的历史沿革，剖析了古典的加密算法为何会被现代的加密算法所取代，展望了密码编码领域的发展，为古典和现代密码体系提供了数学理论基础，还给出了一些针对各种加密算法的密码分析方法。

本书适合作为高校计算机安全与信息安全专业密码学导论的简明教材，也可供对密码学、数论和计算机数论有兴趣的技术人员参考。

Simplified Chinese edition copyright © 2003 by PEARSON EDUCATION NORTH ASIA LIMITED and China Machine Press.

Original English language title: *Making, Breaking Codes: An Introduction to Cryptology* (ISBN 0-13-030369-0), 1e, by Paul Garrett, Copyright © 2001.

All rights reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Prentice-Hall, Inc.

本书封面贴有 Pearson Education（培生教育出版集团）激光防伪标签，无标签者不得销售。
版权所有，侵权必究。

本书版权登记号：图字：01-2001-3870

图书在版编目（CIP）数据

密码学导引 /（美）加内特（Garrett,P.）著；吴世忠等译. —北京：机械工业出版社，2003.8
（计算机科学丛书）
书名原文：Making, Breaking Codes: An Introduction to Cryptology
ISBN 7-111-12478-2

Ⅰ.密… Ⅱ.①加… ②吴… Ⅲ.密码—理论 Ⅳ.TN918.1

中国版本图书馆 CIP 数据核字（2003）第 050100 号

机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码 100037）
责任编辑：[illegible]
北京瑞德印刷有限公司印刷·新华书店北京发行所发行
2003 年 8 月第 1 版第 1 次印刷
787mm×1092mm 1/16 • 27.75 印张
印数：0 001-5000 册
定价：39.00 元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线电话（010）68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专诚为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业

的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzedu@hzbook.com
联系电话：（010）68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周克定	周傲英	孟小峰	岳丽华	范　明
郑国梁	施伯乐	钟玉琢	唐世渭	袁崇义
高传善	梅　宏	程　旭	程时端	谢希仁
裘宗燕	戴　葵			

译 者 简 介

吴世忠：博士、研究员，中国信息安全产品测评认证中心主任。现为全国信息安全标准化技术委员会副主任，中国信息产业商会信息安全产业分会理事长，《信息安全与通信保密》杂志主编。已公开出版文章百余篇，著有《信息系统的互连与互通》、《C3I系统的安全与保密》、《关贸总协定：中国准备好了吗？》、《首都信息化标准指南·信息安全与保密标准化体系》等专著五部和《应用密码学》、《网络信息安全的真相》、《密码学的理论和实践》、《中文 Windows 2000 的安全性》等译著五部，同时还主持起草了防火墙、应用网关安全技术要求以及信息技术安全性评估准则等 7 项国家标准，并主笔撰写了与信息安全战略技术发展有关的多篇专题报告。

宋晓龙：男，1970 年 9 月出生。2000 年 5 月毕业于中国人民解放军信息工程大学，获理学硕士学位；主要研究兴趣为密码算法与理论、密码分析和密码产品测试技术；曾在《通信学报》、《信息安全与通信保密》等刊物发表论文多篇，译著有《密码编码和密码分析：原理与方法》。

郭涛：男，1974 年 9 月出生，湖北宜昌人。毕业于华中科技大学计算机学院，2003 年 10 月将获得工学博士学位；主要研究方向为安全电子支付、信息安全、密码学；曾在《通信学报》、《高技术通讯》等刊物发表论文十几篇。

译 者 序

自从 20 世纪末以来，信息安全成为了人们密切关注的热点问题，而作为信息安全理论基石的密码学更是成为学术界炙手可热的研究方向。市场上关于密码学的译著已经不下几十本，当然也包括我们翻译的《应用密码学》、《密码编码和密码分析》（中译本由机械工业出版社引进出版）等书。而明尼苏达州大学数学系的 Paul Garrett 教授所著的这本书以其深入浅出、条理清晰的风格而在诸多密码学专著中独具特色，它弥补了其他密码学专著忽略密码学相关数学知识的不足，介绍了与密码学相关的几乎所有数学知识。通过阅读本书，读者可以对密码学涉及到的所有数学知识有一个比较全面的了解，有助于加深读者对密码学的理解。

本书并没有深入阐述密码理论和具体的密码算法，而是就密码学相关的数学知识做详细介绍。如果跳过数学知识部分，本书可以作为一本密码学导论。但本书更是一本数论教程，一本以密码学为主线、包含抽象代数的广义数论教程。因此，本书不仅适合于广大数学、密码学专业的学生阅读，也适合于密码学和网络安全专业人士参考。我们希望所有读者阅读之后都能有所收益。

本书的翻译工作受到国家自然科学基金重大项目（90104033）的资助。

本书由吴世忠、郭涛、宋晓龙主持翻译，其他参与翻译校对工作的人员还有杨玉斌、付敏、彭建芬等，在此一并表示感谢。

由于水平所限，翻译不妥或错误之处在所难免，敬请广大读者批评指正。

吴世忠

2002 年 9 月

于中国信息安全产品测评认证中心

前　言

本书主要介绍现代密码学的加密思想及其实现方法，涉及数论、概率论、抽象代数。此外，还有一些是加密算法思想的描述及复杂度理论。我们在讨论安全通信及其理论时，有三个专业术语的含义有些许差异，它们是*密码编码*（cryptography）、*密码分析*（cryptanalysis）和*密码学*（cryptology）。其中，密码编码指的是用各种不同的加密算法对信息进行加密，以保证其安全性；密码分析是相对于密码编码而言的，它包含有密码攻击、发现漏洞和系统安全性证明的内容。密码学则是一个比较综合的概念，它涵盖了密码编码和密码分析两个概念，这也是使用最频繁的一个词。

在介绍了密码编码、密码分析和密码学的概念之后，下面我们介绍一下本书的主要内容：

1) 介绍密码学的历史沿革，特别是给出一些古典的加密算法，并且分析这些算法为什么被现代的加密算法所取代。
2) 密码编码领域的展望（在实际应用领域，加密并不仅仅用于数据的安全和保密）。
3) 介绍古典和现代密码体系的同时，给出这些密码体系的数学理论基础。
4) 给出一些针对各种加密算法的密码分析方法。
5) 阐明密钥管理和执行是基础。

阅读本书要求读者具备跟微积分相关的复杂数学和一些基本的线性代数知识。

首先我们将有选择性地介绍*古典密码学*，这主要是指 20 世纪 40 年代之前的密码编码和密码分析技术。特别是在 1935 年到 1940 年期间，一些通过机械的和初级的电子设备自动实现加密和解密的设备大量出现，这些设备工作速度很慢，而且非常的笨重。这主要是由于加密和解密的过程基本上是用机械和电子方法实现的，并不是通过软件实现的。

按照现代的标准来衡量，那些古典密码（德国的恩格码之前的密码）看来是很失败的。这主要归功于现代的计算机远比 20 世纪 40 年代那些用真空电子管的机器要好得多，而且现在的加密算法远比以前的复杂，主要是为了防止一些在以前看来是不可思议的攻击。

虽说古典的密码分析和现代的密码分析有很大的区别，但它们有一个共同的地方：都使用了数学中的*统计算法*（stochastic algorithm）或者*概率算法*（probabilistic algorithm）。这相对于初等数学中非常传统而且较常用的*确定性算法*（deterministic algorithm）形成了鲜明的对比。对于许多应用目的而言，这样的算法运行速度非常快但成功率却达不到 100%，或者说通常运算速度较快，但不总对。这似乎就像现实生活，而非人为的特意忽略。

在这里需要说明的一点就是，计算机的普及对密码学理论产生了很大的影响，极大地改变了密码学，例如：

1) 加、解密的工作可以自动完成，大规模的加、解密的运算变得很容易完成，设计更为精巧的系统成为可能。
2) 计算机网络中数据的存储、传输和数据处理的激增，对有效的加密及相关技术提出了新的*要求*。
3) 当然，这也使密码分析攻击变得很容易。所以，可能以前只有小孩或者间谍才感兴趣的问题，而现在却有很多人感兴趣。

这是一门应用数学的学科，在以往我们所接触的数学绝大部分都是由应用来推动的。在这里，我们将要接触的有：数论、线性代数、抽象代数、概率论、复杂度理论和其他一些数学知识。当然，我们在本书中不可能把所有的理论一一细述，但会介绍一些和本书相关的知识。

当然，在本书中我们也没有足够的篇幅把有关密码学的历史演变和现在的发展完整地讲述一遍。我们只会给出一些具有代表性而且在密码学发展历史上比较重要的例子，并描述一下密码学发展的其他分支。

我们也不可能全面模拟现实生活中的各种有关加密的问题并进行讨论，密码分析尤其是更不可能。因为我没有实际接触过那些机器，而且，在现实生活中，无论是加密还是密码分析的模拟，都可能需要几个小时甚至几天的时间，此外还需要大量的存储空间。普通的计算机处理加密和（授权的）解密很快，而现实生活中对密码系统的攻击可能需要花费数天甚至数月的时间。

因此，我们会首先讨论一些古典的加密系统，以及这些加密系统中所使用的数学知识，甚至用来理解或者破译这些加密系统的数学知识，这是一种好的热身方式。然后我们会讨论一种现在正在使用的对称加密系统——DES(数据加密标准，Data Encryption Standard)。DES 加密算法比那些古典的加密算法要复杂得多。当然，也正是由于它的成功，目前还没出现一种好的攻击方法。DES 早在 20 世纪 70 年代中期就已经成为美国的一套加密标准(对称密码)。而且，DES 标准除了在美国以外的其他国家也得到了广泛的应用。从 DES 标准被采用至今，经过 20 多年来的密码分析考验，还没有发现它有什么致命的弱点。但现在计算机的运算速度已是远非 1976 年那时候的计算机所能比拟的，通过穷举法进行攻击已经成为可能。事实上，在 1998 年中期，美国的 EFF（Electronic Frontier Foundation）花费了 100 000 美元，用一般商用元件构造了一个 DES 破解器，这个破解器可以在两天内获得一个 DES 密钥。好在还可用 DES 做三次加密，即所谓的 3-DES 加密算法，这种算法被认为是一种比较安全的算法。然而，美国国家标准协会（National Institute of Standard）正在征集一种新的 128 位对称加密算法。迄今为止，征集还没有结束，最终被采纳的加密算法将被命名为高级加密标准(Advanced Encryption Standard，AES)[⊖]。

本书中还将讨论另外一类密码算法——非对称加密算法，又称公钥密码算法。我们将主要讨论两种公钥密码算法：一种是 RSA 算法，另外一种是 ElGamal 算法及其应用。RSA 加密算法比较简单而且也很流行，但 ElGamal 算法更适用于椭圆曲线密码。RSA 加密算法的安全性基于数学中的一个难题：大整数因式分解。ElGamal 加密算法的安全性则是基于这样一个难题：有限域中的离散对数计算（这在后面会给出具体的含义）。这两种密码体制中的任何一种加密算法都需要生成大量且很大的素数，其实这本身就是一个有趣的问题。在介绍完这两个加密算法后，我们将进一步介绍 NTRU 密码，这是一种新的密码，从数学理论上它显得更为成熟。与对称密码体制相比，非对称密码体制由于它更加依赖于数学理论的本质，似乎看上去更容易受到攻击。在这部分我们将介绍一些重要而且精妙的数学问题。

在介绍完经典问题以后，我们会专门给出一些数论的知识，这些知识和现代加密系统有

⊖ 2000 年 10 月，美国国家标准技术局选定由比利时人 Vincent Rijmen 和 Joan Daemen 设计的 Rijndael 加密算法为 AES 的惟一候选算法。2001 年已确定为美国联邦信息处理标准 FIPS190。——译者注

着很大的关系，在公钥加密体制（如 RSA 加密系统和 ElGamal 加密系统）中尤其如此。这部分将包括下面这样一些内容：

1) 公钥（非对称）密码
2) 伪随机数发生器（pRNG）
3) 协议

必要的数学知识将包括：

1) 一些数论和抽象代数的结论
2) 素性检验、因式分解及相关算法
3) 复杂度理论

我们不会过多地讨论复杂度理论，在书中我们只简单地介绍一下复杂度理论的有关指标，并且判断一些问题的复杂程度。

在这里，素数检验和整数分解是所有问题的根本，许多实际的加密算法都可以通过这些基本问题来描述。尽管有时候要解释一个完整的算法，往往还需要很多其他的知识。但是不用解释算法的实质，我们也有可能通过实验对算法的性能和准确性有一个直观的认识。

首要的基本问题就是整数模 n 的结构以及它的一般化问题，记为 $\mathbf{Z}/n$。我们需要明白，对于 n 为合数和 n 为素数时，得到的这个 $\mathbf{Z}/n$ 存在本质上的区别。

在好多高效的算法中，还有一个很重要的随机化问题。在数学中，我们可能已经习惯了每个问题都会有一个肯定的结果，即确定性。随机化可能看上去不是那么容易理解，但在许多情况下，这又恰恰是很多加密算法的重要的理论基础。那么我们直接面临的问题就是去考虑概率意义上的素性检验，比如索洛维-斯特拉森方法和米勒-罗宾方法，并证明它们真正可行。

本书中所包含的内容，可能已经超过了一学期的课程，目前书中的内容都是经过我精心筛选的。如果给一年的时间来学习这门课，可能就会比较轻松一点。

本教程在实际教学中已经使用了多次，并且都是基于这么一个前提，学生对数论、抽象代数、概率甚至密码编码都没有什么了解。密码应用总是离不开数学问题，本书则使得注重实用的人和注重理论的人都可在书中得到各自感兴趣的信息。在编写此书的过程中，我已经尽量使各章节内容相互独立，以便读者可以跳过自己不喜欢的章节，而不影响对其他章节的理解。因此，在某些章节，可能会重复出现某些知识。从教与学的观点看，适当的内容重复是一件好事。

如果把此书作为一学期的数论教材，可以跳过密码编码和计算部分，但可把这部分作为选读资料。当然，在本书中，还有很多抽象代数的知识，这些知识已经远远超过了传统的数论教程。在我为本科生讲授数论课程的时候，我总是考虑一个问题，那就是，在讲授数论的时候是将抽象代数的知识作为独立的预备知识，还是由数论的知识来引出抽象代数的知识。最后我往往会选择后者，就是在讲述数论知识的同时，我一般会附带上抽象代数，但很少会有一本教材能够符合这个要求。本书的一部分内容是我为本科生所写的讲义，在那个讲义中我将数论和抽象代数都纳入其中，并将数论作为引入抽象代数的一个具体入口，而反过来数论的一些基本结论又来源于抽象代数。在选择把此书作为数论的教程时，应该跳过前面六章的内容（前六章主要讲的是加解密的问题）。此外，还要跳过“希尔密码”这一章。有关公开密钥加密系统的章节也可以跳过，但这却是数学在通信中的主要应用之一。

此外，可以将此书作为密码学知识的简短介绍性教程（略过有关数学的知识）。为了使本书更容易理解，书中在涉及到数学知识的时候，一般只提到一些必须需要了解的知识，而且都是一些基本知识。我在编写时也力图使这些数学知识无论是从初级的观点，还是高级的观点都是容易理解的。一般来说，在遇到需要证明的时候，我们往往都对特殊情况给出初等证明，而对一般情况给出较高层次的证明。我认为在教学中，这是一种比较好的教学方法，而且我也没有在这方面吝惜时间和篇幅。另外。一些比较严格的密码学教材都有一个通病，即相关的数学知识受到了冷遇。还有一个普遍的局限就是它们总是假定读者已经具备了相当的数学功底。相比较而言，在本书中不仅为希望了解数学在密码学中如何应用的学生提供了充足的数学资源，而且还尽可能降低对数学知识的要求。因此，本书完全可以作为一本密码学导论的简明教材。从某种意义上说，这也是写作本书的初衷。

如果要把此书作为计算数论的教材，我建议读者应该把注意力集中在算法上，减少对密码学以及更多的理论数学部分的关注。在我教授这门课的过程中，我没有假定学生能够或者愿意做任何的计算机的工作，毕竟这需要 CPU 的时间。在本书中，我详细地给出了算法的描述，目的就是使其清晰易懂，但并没有给出具体的算法的伪代码或者算法的某种语言实现。我之所以这么做，很重要的一个原因就是，我希望学生至少明白算法是如何工作的，而不是简单地去执行它。我没有用专用语言写出算法的另一个原因，是因为我无意认可某种语言及其所需要的全部东西。尽管我坚决支持学生去学习如何编写程序，但不鼓励他们去研究软件包。但是，界面友好的软件包的确很容易上手。

在授课的过程中，若有些学生已经学习过概率论或者数论的知识，就可以跳过其中一些相关的章节。在本书的编写工作中，我将大量的数学知识放到了各章各节中去讲解，而不像有的教程将所有的知识放到最后的附录中。我是基于这么一种考虑，那就是，已经学过这些知识的学生可以跳过相关章节不看，而不了解这些知识的学生可以直接按着顺序阅读，无需为了看一些相关知识而前后翻个不停。这样组织也是为了各章保持独立性。

最后，我要感谢我的朋友们，他们给我的初稿提出了好多宝贵的意见，在此我特别感谢：美国艾奥瓦州立大学的 Irvin Roy Hentzel 教授、艾奥瓦大学的 Yangbo Ye、圣母玛利亚大学的 Joachim Rosenthal、密苏里大学的 Daniel Lieman、密歇根州立大学的 Jonathan Hall 等人。在我写书的这么多年中，我的学生们一直都在使用这些初稿，我要感谢这些学生，他们给我提出了很多好的建议，而且也指出了初稿中的若干问题，使本书的质量有了较大的提升。

Paul Garrett
于明尼阿波利斯，明尼苏达大学
garrett@math.umn.edu
paul.garrett@acm.org
http://www.math.umn.edu/~garrett/

引　言

密码技术的应用：以前，密码技术的最直接的目的其实就是保密。通常，保密的意思就是用加密算法将有用的消息加密，使得即便敌人窃听、截获那些加密后的消息，也很难知道原先的内容是什么。而对于那些既定的或授权的用户，获得加密消息后，可以很容易地解密出原始消息。

目前密码学的一个新应用是消息认证，而不论这个消息是否需要保密。认证的意思是，消息的接收者需要通过一种方法来确认这是合法的发送者发来的信息，或者合法的发送者发来的消息是不是已经被修改了。验证消息在中途是不是已经被修改了非常重要，这一问题也称作数据的完整性。在传输信息的过程中，如果只考虑环境中有噪声影响，而不考虑有人在窃听，则一个相对简单的校验和就足够了。但如果同时考虑到有人破坏，则要实现此功能就会比较复杂，因为设备还要检测是否存在欺骗。

此外，目前还有一种比较流行的应用，那就是签名，以前的签名都是用钢笔来签，这一般称作物理签名，而在网络通信中，这种非物理的签名叫做数字签名。

密码技术另一个不太明显的应用就是非否认问题，尽管这是签名的一个方面：必须让数字签名的签署者事后否认对此签过名成为不可能。这就引起了一个识别欺骗的问题，我们能否使欺骗变得不可能，或者使欺骗可以被检测到？后者可能更容易实现，尽管在一些场合这是不可接受的。

不经意传输：不经意传输指的是如果 Alice 传输一个秘密给 Bob，此后，Alice 就不知道 Bob 是否接收到了这个秘密（但 Bob 知道他是不是收到了）。或者还有一种可能就是，Alice 卖给 Bob 几个秘密中的一个，使得 Alice 不知道 Bob 到底买的是哪一个秘密。

例如在政治条件下，Bob 承认忽略了某些秘密是件尴尬的事情。在这种环境下，信任裁判的简单方案是不可行的。

零知识证明：零知识证明指的是这么一种协议标准，Alice 可以证明给 Bob 看，她知道一个秘密，并且不会泄漏这个秘密。一般来说，当 Alice 证明她知道秘密时，最小泄露证明将使 Alice 告知的信息极小化。欺骗应当是困难的：如果 Alice 根本就不知道秘密，那么她能够使 Bob 错误地相信她的概率应当可以忽略，并且 Bob 不能从 Alice 的证明中获得更多的信息。特别地，Bob 应当不能够（失败）向任何人证明他知道秘密。（最小泄露证明是可能的，数学或其他技术课程的学生可以联想到，他们十分确信讲课者确实知道如何证明一个定理，即便授课者没有给出更多的信息。）

术语：密码体制或密码指的是一个完整的过程，通过这个过程的处理，可以使得一些明文信息，对于授权用户来说，很容易理解，而对于未授权的用户，就显得很难理解。由发送者执行的**加密处理**，指的是发送者将一些明文信息通过加密算法，加密成一些对于未授权用户不可理解的信息。由合法的既定接收者执行的**解密处理**，指的是通过解密算法和密钥，把难解的信息（**密文**）恢复成原始信息（**明文**）。注：对于没有授权的用户，想通过密文而获得明文会非常的困难。在对称加密系统中，应该只有发送者和接收者**共享一个秘密**，称为**密钥**。不让窃听者知道密钥，一般来说可以很有效地阻止他们破解密文。

你现在或许会认为使所有的加密和解密处理都保密，就可以很好地保证加密系统的安全性。然而，事实并非如此，你是不是会觉得不可思议？但事实就是如此，从现在加密安全的观点来看，加密和解密算法都可以公开，只要保证加密密钥保密。这个观点有时被称为**Kerckhoff 原则**，对这个原则有几个不同的看法。其中最引人注目的一个看法很简单，就是不必担心别人知道加密过程，而且设计出符合这个要求的密码体制是可能的，那么不满足这一要求的任何体制都将是不能接受的。

一种与 Kerckhoff 原则一致的看法认为，算法的标准化使得实现大规模的通信变得更加容易。我们可以这么认为，这个加密系统的工作过程对大家来说是公开的。这并不意味着就没有什么需要保密的，唯一的**秘密**就是**密钥**。因为，**密钥分配**和**密钥管理**就成为了一个很重要的问题。

尽管在通俗英语中，“code”（代码，编码）和“cipher”（密码）是同义词，但我们对它们的使用却是不同的。**代码**是利用某个字典，将英语中的词或短语变换为另一种词或短语，从而隐藏消息内容的一种方法。这种变换通常在某种程度上依赖于消息的语法和含义：比如名词和动词被识别并被转换为名词和动词。相比而言，**密码**则把消息当作**字符**流，（一个字符可能是一个字母、数字、空格或者标点符号）不去参照其任何可能的含义。还有一个类似的词，“encode”（**编码**），它指的是非隐藏目的的变换，但却是为了后来的处理更加方便。比如，在对英文加密处理之前，先要将 26 个字母 a~z 编码为数字 0~25。

古典密码学：这指的是在电子计算机出现以前已经得到应用或者已经发明的密码学理论。在二次大战中，有很多恩格码和其他机器用于实现这些加密算法。其实在那时候，那些加密系统及其加密算法已经有了很大的改变，它们的性能已经得到了很大的提高。所以，这里说的“古典”的意思，在很大程度上指的是那些相对于现代系统来说不是很好的系统（用现在的标准衡量）。

对称加密系统：这指的是在加密系统中，加密密钥中含有解密密钥的信息，或者更极端的是在系统中，加密密钥和解密密钥完全一样。这样，信息的发送者和接收者共享同一个密钥，所有的古典加密系统都是这种对称加密系统。事实上，1975 年以前，一直都只有这种加密的方法。但在 1975~1978 年的时候，Merkle 和 Hellman 提出了一种新的加密系统：**非对称密码**，在这种加密系统中，加密密钥含有很少的解密密钥的信息，反之亦然。虽然这种加密系统看上去不太可靠，但实际上它是在许多数学难题的理论基础上建立起来的，所以它的安全性可以由数学知识加以证明。而对称密码则不像非对称密码那样神秘（最近已经知道，在 Merkle 和 Hellman 提出非对称密码思想的 10 年前，英国秘密机构的研究人员已经提出了公钥密码的思想）。

对密码系统的**攻击**：指的是在不知道密钥的情况下，对一个加密系统进行攻击，以获取明文信息。一般有四种基本攻击类型：

1) **唯密文的攻击**：在这种情况下，密码分析者得到了密文片断，但是不知道任何明文，也不知道密钥。这种攻击一般又分为两种情况：一是只解密某个特殊的消息；另一种情况是，获得了密钥，然后进一步破解所有后面的加密消息。
2) **已知明文攻击**：在这种情况下，密码分析者得到了全部或部分明文，以及这些明文所对应的密文，目的是通过这些信息来判断密钥。大多数的古典加密系统最容易被这种攻击方法攻破。

3) **选择明文攻击**：在这种情况下，密码分析者可以选择一定数量的明文，并分析对应的加密，攻击目的是获得密钥。大多数的古典加密系统也很容易被这种攻击方法攻破。

4) **对加密密钥的攻击**：这是对那些由加密密钥的信息容易得到解密密钥信息的非对称加密系统而言的。目的是可以预先处理，在截获任何密文前得到解密密钥。

对攻击方法的分类并不能像其字面意思那样明确。通常，攻击者不知道整个明文，但由于特定环境的原因，攻击者可以肯定某些特定的词会在明文中出现。一个被认为会在明文的某个地方出现的词称为一个**明密对照**（crib）。直到第二次世界大战，对这种明密对照的熟练使用仍在密码分析中发挥了重要作用，但从那以后情况就有所不同了。

前面三种攻击方法是对应于对称加密系统的。在这三种攻击方法中，显然对密码分析者而言，唯密文的攻击是最难的一种攻击方法，而选择明文攻击则是最容易的一种方法。当然，任何一种加密系统都必须能够对抗唯密文攻击，因为对消息的窃听、截取是司空见惯的事情。通常，要想实施已知明文攻击或者选择明文攻击是很困难的，尽管仍有不少对这种情况老生常谈的描述。不论这种攻击的可信度如何，确实有能够对抗选择明文攻击的密码系统。结果是能否对抗选择明文攻击成为衡量一个加密系统的标准。因此，一些古典密码系统是由于不能对抗唯密文攻击而被废弃的，而事实上由现在的标准来衡量，因为它们不能对抗选择明文攻击而应当被坚决拒绝。我们还要考虑对古典密码可能实施的更加困难的唯密文攻击，这既是我们本来的目的，也是要表明一些有趣的问题。当然我们还会指出它们对已知明文攻击和选择明文攻击的脆弱性。读者应当理解，目前的标准就是：*一个密码系统应当能够对抗选择明文攻击*。

有一个词**试探性**（heuristic）我们会经常用到。对一个问题的试探性方法决不保证其是否工作或者是否正确，但却对解决问题的步骤或者什么是正确的提出了建议。在最佳情况下，这种方法可以得到验证。

术语**算法**是一个能够完全自动化（比如在计算机上编程）的计算或决策程序。一个**概率算法**则是一个不能总是正常工作或者不可能得出一个保证正确结果的算法。

我们还要指出，对于信息的度量，采用比特和字节等度量单位。一个**比特**是一个最小的度量单位，0 或 1（真或假、开或关）。有些文章中认为比特是"二进制位"的缩写，可能这也是正确的。一个**字节**指的是 8 个比特的串，但其中一位通常用作校验位（用于校验任何比特是否有错误），因此，一个字节传输的信息种类是 2^7 种而不是 2^8 种。此外 **1KB**（kilobyte）表示 1000 字节，**1MB**(megabyte)表示 1 000 000 字节，**1GB**(gigabyte)表示 1 000 000 000 字节的意思，**1TB**(terabyte)表示 1 000 000 000 000 字节等等。

在使用拉丁字母（A 到 Z 不包括发音符号）的国家，通常一个字符被编码为 0~255 之间的一个整数，即一个字节。这当中有许多是不可打印的字符或控制字符。当然还有其他字符体系，比如美国的 **ASCII 体系**是比较常用的：字符 0~9 的编码为 48~57，大写字母 A~Z 的编码为 65~90，而小写字母 a~z 的编码为 97~122，其他整数编码表示标点符号和控制字符。

Unicode 是一种针对较大字母表（象形文字系统，如汉字）的新型编码方法，通过使用两个字节来表示一个字符，这就可以提供 2^{16}=65 536 种字符编码，比单字节的 256 种字符编码有了很大的进步。

目　录

出版者的话
专家指导委员会
译者简介
译者序
前言
引言
第 1 章　简单密码 …… 1
1.1　移位密码 …… 1
1.2　约简/整除算法 …… 4
1.3　一次一密密码本 …… 7
1.4　仿射密码 …… 9
第 2 章　概率 …… 13
2.1　计数 …… 13
2.2　基本思想 …… 15
2.3　英文统计 …… 23
2.4 对仿射密码的攻击 …… 28
第 3 章　置换 …… 31
3.1　暗号：代替 …… 31
3.2　变位字：换位 …… 33
3.3　置换概念 …… 37
3.4　洗牌 …… 42
3.5　分组交错 …… 43
第 4 章　严格的密码 …… 45
4.1　维吉尼亚密码 …… 45
4.2　最小公倍数 LCM 和最大公约数 GCD …… 48
4.3　Kasiski 攻击 …… 49
4.4　期望值 …… 54
4.5　Friedman 攻击 …… 57
第 5 章　概率问题 …… 71
5.1　生成函数 …… 71
5.2　方差、标准差 …… 73
5.3　车贝雪夫不等式 …… 74
5.4　大数定律 …… 75
第 6 章　现代对称密码 …… 77
6.1　设计目标 …… 77
6.2　数据加密标准 …… 79
6.3　高级加密标准 …… 84
第 7 章　整数 …… 87
7.1　整除性 …… 87
7.2　因式唯一分解 …… 89
7.3　欧几里得算法 …… 94
7.4　乘法逆元 …… 97
7.5　乘法逆元的计算 …… 99
7.6　等价关系 …… 101
7.7　整数模 m …… 103
7.8　本原根和离散对数 …… 107
第 8 章　希尔密码 …… 111
8.1　希尔密码原理 …… 111
8.2　对希尔密码的攻击 …… 112
第 9 章　复杂度 …… 119
9.1　大 O 和小 O 符号 …… 119
9.2　位操作 …… 120
9.3　概率算法 …… 123
9.4　复杂度 …… 123
9.5　子指数算法 …… 124
9.6　柯尔莫哥洛夫复杂度 …… 125
9.7　线性复杂度 …… 126
9.8　最差情况与期望值 …… 126
第 10 章　公钥密码算法 …… 129
10.1　陷门 …… 130
10.2　RSA 密码 …… 131
10.3　Diffie-Hellman 密钥交换 …… 137
10.4　ElGamal 密码 …… 138
10.5　Knapsack 密码 …… 141
10.6　NTRU 密码 …… 143
10.7　算术密钥交换 …… 146
10.8　量子密码 …… 149
10.9　美国出口限制 …… 151
第 11 章　素数 …… 153
11.1　欧几里得定理 …… 153

11.2 素数定理 ····· 153
11.3 序列中的素数 ····· 154
11.4 车贝雪夫定理 ····· 155
11.5 最佳渐进法 ····· 157
11.6 黎曼假设 ····· 158
第 12 章 mod p 的根 ····· 159
12.1 费马小定理 ····· 159
12.2 特殊的因式分解表达式 ····· 160
12.3 梅森数 ····· 161
12.4 更多的例子 ····· 163
12.5 指数算法 ····· 165
12.6 mod p 的二次根 ····· 167
12.7 mod p 的高次根 ····· 168
第 13 章 模合数的根 ····· 171
13.1 孙子定理 ····· 171
13.2 特殊方程组 ····· 173
13.3 模是合数的同余方程 ····· 175
13.4 亨泽尔引理 ····· 177
13.5 平方根 oracle ····· 180
13.6 欧拉定理 ····· 182
13.7 原根的性质 ····· 183
13.8 欧拉判别准则 ····· 184
第 14 章 弱乘法性 ····· 187
14.1 弱乘法性的定义 ····· 187
14.2 算术卷积 ····· 188
14.3 墨比乌斯反演 ····· 190
第 15 章 二次互反定理 ····· 193
15.1 二次根 ····· 193
15.2 二次符号 ····· 194
15.3 乘法性质 ····· 194
15.4 二次互反律 ····· 195
15.5 快速计算 ····· 199
第 16 章 伪素数 ····· 203
16.1 费马伪素数 ····· 203
16.2 非素的伪素数 ····· 205
16.3 欧拉伪素数 ····· 206
16.4 索洛维-斯特拉森检验 ····· 208
16.5 强伪素数 ····· 208
16.6 米勒-罗宾检验 ····· 209
第 17 章 群 ····· 211
17.1 群概念 ····· 211
17.2 子群 ····· 212
17.3 拉格朗日定理 ····· 213
17.4 子群的指标 ····· 215
17.5 指数定律 ····· 215
17.6 循环子群 ····· 217
17.7 欧拉定理 ····· 218
17.8 群的指数 ····· 218
第 18 章 协议概述 ····· 221
18.1 基本的公钥协议 ····· 221
18.2 Diffie-Hellman 密钥交换 ····· 222
18.3 秘密共享 ····· 223
18.4 不经意传输 ····· 224
18.5 零知识证明 ····· 226
18.6 鉴别 ····· 226
18.7 电子货币和电子商务 ····· 228
第 19 章 环、域、多项式 ····· 231
19.1 环、域 ····· 231
19.2 整除性 ····· 235
19.3 多项式环 ····· 236
19.4 欧几里得算法 ····· 237
19.5 欧几里得环 ····· 240
第 20 章 分圆多项式 ····· 245
20.1 特征 ····· 245
20.2 重因子 ····· 246
20.3 解分圆多项式 ····· 249
20.4 本原根 ····· 251
20.5 模 p 的本原根 ····· 251
20.6 素数方幂 ····· 252
20.7 本原根的计数 ····· 254
20.8 不存在性 ····· 255
20.9 搜索算法 ····· 256
第 21 章 随机数发生器 ····· 257
21.1 假的一次一密乱码本 ····· 257
21.2 伪随机数发生器的周期 ····· 258
21.3 同余发生器 ····· 258
21.4 反馈移位发生器 ····· 260
21.5 Blum-Blum-Shub 发生器 ····· 261

21.6 Naor-Reingold 发生器 ····· 262
21.7 线性同余发生器的周期 ····· 263
21.8 本原多项式 ····· 265
21.9 线性移位寄存器的周期 ····· 267
21.10 本原多项式的例子 ····· 269
21.11 本原性检验 ····· 271
第 22 章 群的更多知识 ····· 275
22.1 群同态 ····· 275
22.2 有限循环群 ····· 277
22.3 无限循环群 ····· 280
22.4 群中的根和方幂 ····· 280
22.5 平方根算法 ····· 282
第 23 章 伪素性证明 ····· 287
23.1 λ函数 ····· 287
23.2 卡米克尔数 ····· 288
23.3 欧拉证据 ····· 289
23.4 强证据 ····· 291
第 24 章 因式分解攻击 ····· 297
24.1 Pollard 的 Rho 方法 ····· 297
24.2 Pollard 的 $p-1$ 方法 ····· 300
24.3 Pocklington-Lehmer 准则 ····· 302
24.4 强素数 ····· 306
24.5 素性证书 ····· 308
第 25 章 现代因式分解攻击 ····· 313
25.1 高斯消元法 ····· 313
25.2 随机平方分解 ····· 315
25.3 Dixon 算法 ····· 315
25.4 非筛的二次筛法 ····· 317
25.5 二次筛法 ····· 319
25.6 其他改进 ····· 319
第 26 章 有限域 ····· 321
26.1 有限域的构造 ····· 321
26.2 域扩张的例子 ····· 322
26.3 模 P 加法 ····· 323
26.4 模 P 乘法 ····· 324
26.5 模 P 乘法逆 ····· 324
第 27 章 离散对数 ····· 327
27.1 Baby-step Giant-step 算法 ····· 327
27.2 Pollard 的 Rho 方法 ····· 329
27.3 指数演算 ····· 334
第 28 章 椭圆曲线 ····· 337
28.1 抽象的离散对数 ····· 337
28.2 离散对数 ····· 337
28.3 椭圆曲线上的运算 ····· 339
28.4 无穷远点 ····· 343
28.5 射影椭圆曲线 ····· 345
第 29 章 有限域的更多知识 ····· 347
29.1 交换环上的理想 ····· 347
29.2 环同态 ····· 350
29.3 商环 ····· 352
29.4 极大理想和域 ····· 353
29.5 域扩张的更多知识 ····· 354
29.6 费罗贝尼乌斯自同构 ····· 355
29.7 不可约多项式的计数 ····· 360
29.8 本原多项式的计数 ····· 362
附录 A 相关公式 ····· 365
附录 B 部分习题答案 ····· 379
附录 C 常用数表 ····· 383

第1章 简单密码

本章介绍三种相关的简单密码：移位密码、一次一密密码本以及仿射密码。移位密码和仿射密码的性能并不好，而“如果”使用恰当的话，一次一密密码本是相当安全的。但是这里的“如果”也得画一个大大的问号。

我们将从这三种密码着手，给出密码描述的一般模式，随后将分别讨论它们各自的缺陷。同时也会对相关的数学思想作介绍。

讨论弱密码的目的之一是让我们了解它的弱点，以便在未来工作中避开其弱点，同时在需要的情况下知道如何加以使用。很显然，如果我们只看到那些不可攻击的密码，对于密码如何失效就不得而知了，更别提攻击者的一次成功攻击了。因此尽管移位密码看起来很“愚蠢”，还是值得我们去研究一下它的细节，以便给我们提供一些经验教训。

1.1 移位密码

移位密码是最简单的一种密码。这至少要追溯到 Julius Caesar（恺撒，古罗马统帅，公元前 100 年—公元前 44 年。——编者注），因为这种算法有时也被称为 Caesar 密码。在这之前还有更早（而且更好）的密码（如 Polybios 密码）。尽管这确实是一种差劲的加密系统，然而去研究一下它如何差劲也是很有意义的。

首先我们描述 Caesar 密码显而易见的一种特殊情形，然后描述一般的加密方案。我们约定在以后的讨论中，**明文**（plaintext）将用小写字母书写，而**密文**（ciphertext）则用大写字母书写。同时使用 26 个英文字母通常的顺序。所有消息均用英文表示。

在使用 Caesar 密码的情况，为了给消息加密，消息中每个字母均被前移三位，例如“a”将变成“D”，“b”将变成“E”，“c”将变成“F”等。在字母表的最后，移位将重新折回，形成一个循环，“x”将变成“A”，“y”将变成“B”，“z”将变成“C”。

要解密 Caesar 密码系统，所有字母将被后移三位，“A”将变成“x”，“B”将变成“y”，“C”将变成“z”，“D”将变成“a”，“E”将变成“b”，等等。

按照上述 Caesar 密码的加密规则，如下消息

all of gaul is divided into three parts

将变成

DOO RI JDXO LV GLYLGHG LQWR WKUHH SDUWV

这样，即使消息被中途截取，面对大多数既不会读也不会写这句话的人而言，这种方式也还是能起保密作用的。然而，当面对一个熟知该加密系统的攻击者时，破译截取消息的唯一障碍就只剩下推算实际移位值所花费的时间。由于字母移位解密是由接收方完成，万一加密方法泄密，对攻击者来说解密就变得跟接收方解密一样容易。这是我们不希望看到的。

我们可以通过加进一个附加秘密来改进上述方法，使得攻击者必须付出比接收方大得多的工作量才能解密，也即是消息的发送方和接收方事先协商好一个**密钥**。该密钥代表在当前

环境下字母移位的位数（从 1 到 25），而不像前面 Caesar 加密那样所有情况下都向前和向后移动 3 位。该密钥是发送方和接收方事先达成一致的关于那条消息的共同秘密。这种加密系统被称为**移位密码**。

因此，一个幼稚的攻击者在截取一条被移位加密的消息时，就会因缺少信息而不知道如何下手。相反，消息接收方则可以很快通过移位恢复出消息明文。

而一个不轻易放弃的攻击者可能会尝试各种可能的移位，并等着看哪一条最可能是原文。这就是一种**穷举攻击法**。例如，假设攻击者截获到如下消息：

RCC FW XRLC ZJ UZMZUVU ZEKF KYIVV GRIKJ

他将连续尝试将消息中字母分别后移 1 位、2 位、3 位等等，直到消息原文出现：

（后移 01 位）：qbb ev wqkb yi tylytut ydje jxhuu fqhji
（后移 02 位）：paa du vpja xh sxkxsts xcid iwgtt epgih
（后移 03 位）：ozz ct uoiz wg rwjwrsr wbhc hvfss dofhg
（后移 04 位）：nyy bs tnhy vf qvivqrq vagb guerr cnegf
（后移 05 位）：mxx ar smgx ue puhupqp uzfa ftdqq bmdfe
（后移 06 位）：lww zq rlfw td otgtopo tyez escpp alced
（后移 07 位）：kvv yp qkev sc nsfsnon sxdy drboo zkbdc
（后移 08 位）：juu xo pjdu rb mrermnm rwcx cqann yjacb
（后移 09 位）：itt wn oict qa lqdqlml qvbw bpzmm xizba
（后移 10 位）：hss vm nhbs pz kpcpklk puav aoyll whyaz
（后移 11 位）：grr ul mgar oy jobojkj otzu znxkk vgxzy
（后移 12 位）：fqq tk lfzq nx inaniji nsyt ymwjj ufwyx
（后移 13 位）：epp sj keyp mw hmzmhih mrxs xlvii tevxw
（后移 14 位）：doo ri jdxo lv glylghg lqwr wkuhh sduwv
（后移 15 位）：cnn qh icwn ku fkxkfgf kqvq vjtgg rctvu
（后移 16 位）：bmm pg hbvm jt ejwjefe joup uisff qbsut
（后移 17 位）：all of gaul is divided into three parts
（后移 18 位）：zkk ne fztk hr chuhcdc hmsn sgqdd ozqsr
（后移 19 位）：yjj md eysj gq bgtgbcb glrm rfpcc nyprq
（后移 20 位）：xii lc dxri fp afsfaba fkql qeobb mxoqp
（后移 21 位）：whh kb cwqh eo zerezaz ejpk pdnaa lwnpo
（后移 22 位）：vgg ja bvpg dn ydqdyzy dioj ocmzz kvmon
（后移 23 位）：uff iz auof cm xcpcxyx chni nblyy julnm
（后移 24 位）：tee hy ztne bl wbobwxw bgmh makxx itkml
（后移 25 位）：sdd gx ysmd ak vanavwv aflg lzjww hsjlk

所有可能的移位如上所示，通常只有一条消息有意义。我们注意到，攻击者花费了 20 多倍的工作去解密，而接收者事先就知道需要移位的位数为 17。如果消息很长，攻击者将需要花费比 20 倍更多的工作量，这将是不能允许的。

稍微聪明一点的攻击者会意识到，如果连几个初始的短语都不能看懂的话，把所有的消息移位将是很愚蠢的。也就是说，我们不需要将整条消息均作移位解密才能看出它是否有意

义。实际上，看看在上述的 25 种解密文中只看前 3 个字母，只有很少几个能构成英文单词或是单词的一部分。即使前 3 个字母看起来可能是某个单词的一部分，继续尝试着对紧接着的 2 个字母解密后就会发现只有一个是可能的候选明文，这也就是真正的明文。

为了加快穷举攻击的速度，攻击者最多只须利用猜想的密钥对 5 个字符进行解密，就可以判断该密钥是否正确。因为只有 25 种可能，故只需做很少的工作就可以找到移位密码的密钥。

也就是说，移位密码很容易被*唯密文攻击方法攻击*：任何由 3 个字母以上英文组成的消息只需要对前面几个字母进行 25 次不同移位的尝试就可以破译。因此如果每次考察 5 个字母，在找到真正的密钥前就会有 24×5 次尝试被浪费。如果有一份每行 80 个字符共 10 行的消息，攻击者只须花费比接收方多 15%的工作量就可以破译密码，这是挺糟糕的事情。

更糟的出现在*已知明文攻击*：如果攻击者知道一个字符以及它的密文，那么他很快就可以通过加密字符与解密后的明文之间的间距推出密钥是多少（1~25 之间的一个整数）。（选择明文攻击在这种情况下并不会更简单，但如何得到更简单的攻击方法呢？）

尽管移位加密在恶意攻击者面前很难保密，但还是有一些用处的。这种加密虽然脆弱，但对明文进行了不透明的封装，从这种意义上讲，它防止了消息明文被*人意外的获取*。也就是说，尽管它在*主动攻击者*面前基本上不能保密，但在*合作代理*(cooperating agent)面前还是完全有效的。总之，从商业观点来看移位密码很容易被攻破，但很少有人不费力就能自然地解读出加密的消息。因此，谜语的答案可以通过移位加密而不让人很容易就直接看出来，质疑性的玩笑也可以使用移位加密，使无意看到它的人不会有罪恶感。

术语：移位密码是**对称的**，也即在给出解密密钥的情况下就能知道*加密密钥*，反之亦然。它是**单表代替的**，对所有的字母都是按照同样的方法（同一个密钥）进行加密。

习题

1.1.01 用密钥 13 对“this is the message”进行移位加密。

1.1.02 用密钥 7 对“this is the message”进行移位加密。

1.1.03 用密钥 19 对“this is the message”进行移位加密。

1.1.04 用密钥 8 对“this is the message”进行移位加密。

1.1.05 用密钥 12 对“this is the message”进行移位加密。

1.1.06 为 Caesar 密码加密的消息“WKLV VKRXOG EH TXLWH HDVB”解密。

1.1.07 为 Caesar 密码加密的消息“WDNH PH RXW WR WKH EDOO JDPH”解密。

1.1.08 为 Caesar 密码加密的消息“UIFTF BSF UJNFT UIBU NBLF VT UJSFE”解密。

1.1.09 为移位密码加密的消息“VO DOHA H ILHBAPMBS TVYUPUN”解密。

1.1.10 为移位密码加密的消息“YRQ QEFP BUXJMIB FP IBPP BXPV”解密。

1.1.11 为移位密码加密的消息“JYRIB YRJ GIVKKP KVVKY”解密。

1.1.12 为移位密码加密的消息“WX KDBRWNBB URTN BQXF KDBRWNBB”解密。

1.1.13 为移位密码加密的消息“KTBG BG LITBG YTEEL BG MAX IETBG”解密。

1.1.14 为什么说移位幅度大于 25 是不明智的？

1.1.15 为什么向前移位 t 步与向后移位 $26-t$ 是一样的？

1.2 约简/整除算法

对一个非零整数 m，考虑**模 m 约简**（reduction modulo m），也就是用 m 做带余除法。即，对 N **模** m 的**约简**就是 N 除以 m 时产生的余数。这个过程也被称作**整除算法**。

更准确地说，对 N 模 m 的约简就是唯一的整数 r（**余数**），其范围为：

$$0 \leqslant r < |m|$$

因此 N 可以写成：

$$N = q \cdot m + r$$

整数 q 是**商**。我们也可以写成：

$$N\%m = r$$

（通常“模”被简记为 mod）非负整数 m 就是**模数**。例如：

$$10\%7=3\text{（因为 }10=1\cdot7+3\text{）}$$
$$10\%5=0\text{（因为 }10=2\cdot5+0\text{）}$$
$$12\%2=0\text{（因为 }12=6\cdot2+0\text{）}$$
$$15\%7=1\text{（因为 }15=2\cdot7+1\text{）}$$
$$100\%7=2\text{（因为 }100=14\cdot7+2\text{）}$$
$$1000\%2=0\text{（因为 }1000=500\cdot2+0\text{）}$$
$$1001\%2=1\text{（因为 }1000=500\cdot2+1\text{）}$$

为了简单起见，在一些文献资料中通常将“N 模 m 约简”随意地简化为“N 模 m”。但是，我们稍后就会看到“N 模 m”还有另外一个意义相关却又截然不同的含义。通过上下文我们一般都能明白“N 模 m”的确切含义，但是仍要小心以免出错。我们将用一个与大多数计算机语言相当兼容的运算符号来表示：

$$x\%m$$

x 模 m 约简

模 m 运算可以用手工通过熟悉的“**长除**”算法得到。当 N、m 都是**正数**时，即使一个简单的掌上计算器都能用来求模，例如：用 N 除以 m 得到一个数值，减去该数的整数部分，乘回 m 就得到 N 模 m 的结果。另外一种是当模数 m 为大数时仍是比较稳定的算法，是 N 除以 m 得到一个小数，去掉商的小数部分，再乘以 m，最后从 N 中减去上述乘积，得到的就是 N 模 m 的结果。上述两个算法看起来并没有什么区别，毫无疑问计算器的不同就是小数点后所精确到位数的不同。这就是区别，所以第二种算法要好一些，因为当数值大到一定程度时，精确性就成了问题。

上述的模运算同样可以应用到负整数。例如：

$$-10\,\%\,7 = 4 \quad \text{因为} \quad -10 = (-2)\cdot 7 + 4$$
$$-10\,\%\,5 = 0 \quad \text{因为} \quad -10 = (-2)\cdot 5 + 0$$
$$-15\,\%\,7 = 6 \quad \text{因为} \quad -15 = (-3)\cdot 7 + 6$$

但是上面提到的两种算法都不能直接得到正确的结果。因为，首先，$-N$ 模 m 的结果不等于 N 模 m 结果的负数。此外，我们假设所有参与模运算的数都是非负数。例如：

$$10 = 1\cdot 7 + 3$$

表示 10 模 7 余 3，但如果我们将上式两边取负，得到：

$$-10=(-1)\cdot 7+(-3)$$

上式的“-3”并不符合我们的要求。技巧就是将“-3”加上 7 的倍数，同时从（-1×7）中减去该值，得到：

$$-10=(-1-1)\cdot 7+(-3+7)$$

最后，有：

$$-10=(-2)\cdot 7+4$$

还有一个例外，当余数是 0 时，就像：

$$14=2\cdot 7+0$$

当我们取负时，得到：

$$-14=(-2)\cdot 7+0$$

这时，就没有必要再作处理，因为 0 已经是非负数了。（如果我们又加了 7，其结果就会超出正常范围。）总而言之，设 r 是 N 模 m 的余数。那么,如果 $r\neq 0$，则 $-N$ 模 m 的余数是 $m-r$；如果 $r=0$，则余数就是 0。

模数也可以是负数。但是 N 模 m 的运算就是 N 模 $|m|$，所以也就没有必要再介绍了。

注意，这里定义的任何整数模 m 的余数都是非负数，这与一些计算机语言中的用法并不一致。在那里，负整数 $-N$ 模 m 的余数等于 N 模 m 的余数的负数。我们在书写计算机代码时一定要记住这个区别。

整数 N **模** m **的乘法逆**就是一个整数 t，使得 $N\cdot t\%m=1$。认识到新概念中加在“模”前面的“逆”字很重要。因为乘以 t（然后约简 m），最后得到的结果是 1，与倒数有相同的抽象特性。然而，我们知道 2 的倒数 0.5 是有理数或实数，但却不知道 2 模 5 的逆是什么。因此，虽然模 m 的逆与通常意义上的倒数有很强的抽象联系，但是很容易引起误解，误认为有直接切实的联系。

例如，由于 2×3=6，减去模数 5 得到 1，我们可以说 3 是 2 模 5 的逆，但这并不是说，“3=1/2”或“3=0.5”或其他。另外一个例子，143 是 7 模 1000 的逆，因为 7×143=1001，然后模 1000 得到 1。另外，我们可以知道，2 模 10 就不存在乘法逆，因为任何倍数 $2\times t$ 都是偶数，然而 $q\times 10+1$ 肯定是奇数。

因此，作为整数是否有乘法逆并不清楚。对于如何有效地找到乘法逆更是不可知。我们在这里暂且留作一个小小的悬念，但稍后将做出肯定的解答。

最后，我们将证明约简/整除算法中商和余数的存在性和唯一性。

命题 给定一非零整数 m 和任意整数 N，存在唯一的整数 q 和 r，使得 $0\leqslant r<|m|$ 且

$$N=q\cdot m+r$$

证明 对于给定的整数 m 和 N，令 X 是所有形如 $N-x\cdot|m|$ 的整数的集合，设 r 是 X 中最小的非负整数。q 为满足等式 $N-q\cdot|m|=r$ 的整数。

首先，我们证明 $0\leqslant r<|m|$。假设 $r\geqslant|m|$，则 $r-|m|\geqslant 0$。由此 $r-|m|$ 可以写成 $N-(q+1)|m|$,故可知其在集合 X 中。但是 $r-|m|<r$，这与 r 是 X 中最小的非负整数的事实相矛盾。因此 $r\geqslant|m|$ 的假设错误，所以有 $r<|m|$。另外，根据 $N=q|m|+r$，如果 $m>0$，则有 $N=qm+r$；如果 $m<0$；则用 $-q$ 代替 q 仍然可以得到 $N=qm+r$。

下面，我们证明 q 和 r 的唯一性。假设有不同的 q,q' 和 r,r' 满足：

$$qm+r=q'm+r'$$

其中$0 \leqslant r < m$，$0 \leqslant r' < m$。由于对称性，我们可以假设$r \leqslant r'$（反之亦然），得到：

$$(q'-q)\cdot m = r'-r$$

并且$r'-r \geqslant 0$。如果$r'-r \neq 0$，则$q'-q \neq 0$，因此：

$$r'-r = |r'-r| = |q'-q|\cdot|m| \geqslant 1\cdot|m|$$

但是，

$$r'-r \leqslant r' < |m|$$

这样，我们就得到矛盾的式子：

$$|m| \leqslant r'-r < |m|$$

也就是与假设$r \neq r'$矛盾。因此$r = r'$。同样从$(q'-q)m = r'-r = 0$(且$m \neq 0$)，我们可以得到$q' = q$。这就证明了唯一性。

我们还必须确定集合X里面含有一些非负整数,否则我们得出的上述推论就是错误的。这并不奇怪，一方面,如果$N > 0$，则$N - 0\cdot|m|$肯定是正数。另一方面，如果$N < 0$，对于绝对值充分大的负数x，式子$N - x\cdot|m|$是正数。 ♣

备注 任何由正整数组成的非空集合有一个最小元的断言也就是正整数的良序原理(Well - Ordering Principle)。

命题 设n和N为两个整数，对某个整数k有$N = kn$，则对任意整数x

$$(x\%N)\%n = x\%n$$

证明 设$x = Q\cdot N + R$,且$0 \leqslant R < |N|$，这里R为x模N的约简。进一步，设$R = q\cdot n + r$且$0 \leqslant r < |n|$，这里r为R模n的约简，则有

$$x = QN + R = Q(kn) + qn + r = (Qk + q)\cdot n + r$$

因此r也是x模n的约简。 ♣

习题

1.2.01 求1000模99的约简。

1.2.02 求1001模81的约简。

1.2.03 求10模81的约简。

1.2.04 求1000模73的约简。

1.2.05 求1000模89的约简。

1.2.06 求 - 10模88的约简。

1.2.07 求 - 23模67的约简。

1.2.08 求 - 1模123的约简。

1.2.09 求 - 1000模88的约简。

1.2.10 求 - 1模123的约简。

1.2.11 求 - 10模59的约简。

1.2.12 求 - 1000模279的约简。

1.2.13 求 - 997模399的约简。

1.2.14 证明一个正整数N模10的约简就是N的个位数字。

1.2.15 证明一个正整数N模100的约简就是由N的十位数字和个位数字组成的数。

1.2.16 设m是一非零整数，证明N模$-m$的约简与N模m的约简一样。

1.2.17 如果 r 是 N 模 m 的约简，并且 $r \neq 0$，证明 $m-r$ 是 $-N$ 模 m 的约简。

1.2.18 用穷举法求 13 模 100 的逆。

1.2.19 用穷举法求 87 模 100 的逆。

1.2.20 用穷举法求 29 模 100 的逆。

1.2.21 用穷举法求 31 模 100 的逆。

1.2.22 用穷举法验证在 1,2,⋯,25 中具有模 26 乘法逆的整数为奇数，13 除外。

1.2.23 (*) 设 m 是一个正整数，证明对所有的整数 x, y，以下式子成立：

$$((x\%m)+(y\%m))\%m=(x+y)\%m$$

$$((x\%m)\times(y\%m))\%m=(x\times y)\%m$$

1.3 一次一密密码本

一次一密密码（OTP, One-Time Pad）如果被正确使用，则是相当安全的。我们甚至可以证明它是相当安全的，而不仅仅是主观相信。从概念上说它是非常简单的。它是目前世界上唯一已知的与攻击者攻击时间和攻击机器无关的。可保密消息的密码。出于这个原因，世界上一些非常敏感的通信都使用 OTP，比如发射核武器的代码等。OTP 密码有时也称为 Vernam 密码，以 Gilbert Vernam 而得名，此人大约在 1917 年就使用这种密码。下面我们简要描述一下 OTP 密码。

注意这里当然还有一个条件，我们说的是“正确使用”。如果密钥被重复使用，或者使用了质量很差的密钥，则 OTP 密码很容易失败。比如，如果一个密钥被重复使用了，则该密码就成了维吉尼亚密码，稍后我们会看到如何成功攻击这样的密码。

为了使加密和解密的讨论容易进行，我们仍然采用通常的记号，词汇表是模 m 约简的。例如，如果我们将小写字母分别编码为：a—0,b—1,c—2,⋯,z—25，则移位密码的加密就可以简单地用一个由集合 {0,1,2,⋯,25} 到自身的函数 E_k（与密钥 k 确定需要移多少位相关）来给出，具体公式如下：

$$E_k(x)=(x+k)\%26$$

同样，密钥为 k 的移位密码的解密也可看作另外一个函数 D_k：

$$D_k(x)=(x-k)\%26$$

因为移位密码的密钥空间太小（仅为 25），所以它很弱，甚至容易受到唯密文攻击。

我们来考虑另一种极端情况，即设法使密钥与消息本身一样大。这当然是很有帮助的，因为这就产生了一个极大的密钥空间。这就是 OTP 密码的主要思想之所在，当然还要求密钥只能使用一次。由于密钥的一次性使用，这使得 OTP 密码比仅有较大密钥空间的密码又要强一些：无论攻击者花费多少时间和精力，都不会从正确运用 OTP 加密的密文中得到任何信息。这不仅是一个产生无法处理的巨大密钥空间的问题了。

OTP 密码的工作方式如下：假设我们希望发送一个非常重要的长度为 n 的消息，并想通过 OTP 加密。我们选择一个包含 n 个随机字符的密钥 $k=(k_1,k_2,\cdots,k_n)$，这里我们可以将每个字符当作 0 到 25 之间的一个整数（利用前面的编码方法），需要被加密的消息为 $x=(x_1,x_2,\cdots,x_n)$，同样每个字符被编码为 0 以 25 之间的整数。使用加密函数 E_k，它是一个长为 n 的字符串的集合到自己的一个函数。使用以 26 为模进行约简的思路，这个函数的公式

可简写为：x%26（x 模 26 约简）。这样就可将 OTP 加密表示为：

$$E_k(x)=((x_1+k_1)\%26,(x_2+k_2)\%26,\cdots,(x_n+k_n)\%26)$$

也就是说明文 x 中的每一个字符被移位，且移动的位数由对应的密钥来确定。那么解密函数 D_k 同样可表示为：

$$D_k(x)=((x_1-k_1)\%26,(x_2-k_2)\%26,\cdots,(x_n-k_n)\%26)$$

比如，明文长度为 10，密钥是发方和收方事先就要协商好使用一个看起来相当随机的长为 10 的数字序列，每个数均在 0 到 25 之间，假设密钥为 $k=(8,13,24,19,9,1,0,7,20,3)$，明文 x 首先被编码为 $x=(8,12,15,14,18,18,8,1,11,4)$，则

$$\begin{aligned}E_k(x)&=((8+8)\%26,(12+13)\%26,(15+24)\%26,(14+19)\%26,(18+19)\%26,\\&\quad(18+1)\%26,(8+0)\%26,(1+7)\%26,(11+20)\%26,(4+3)\%26)\\&=(16,25,13,7,1,19,8,8,5,7)=QZNHBTIIFH\end{aligned}$$

顺便说一下，OTP 的优点就是它是**多表代替**，即相同的字母依据它在明文中出现的不同位置而用不同的方式加密。这比**单表代替**（移位密码）有明显的优势，因为在单表代替中同一个字母总是被加密成同一个密文字符。

下面我们简要地描述一下 OTP 密码是**相当安全**的证明。首先我们来说明**相当安全**的含义是什么：即截获了密文的攻击者，不会获得关于明文的任何额外的信息，即与他们没有获得密文时的情况一样。当然这听起来就有滑稽，但如果你仔细考虑一下，你会完全赞同这个定义的。那么 OTP 密码是如何达到这一目的呢？粗略的思想描述如下：

随机的 n 比特+有意义的 n 比特 = 随机的 n 比特

也就是

随机的 0 或 1+有意义的 0 或 1 = 随机的 0 或 1

即这里要求随机数资源要与消息一样多，否则随机数就无法完全掩盖消息的含义。

但是，这里仍存在弱点。首先，这里有一个诱惑，即发送方和接收方愿意使用他们容易记忆的词或短语作为他们的密钥（当然要编码为 0 到 25 之间的整数），比如某个人的名字，宠物的名字或者书的标题等等。那么对发送方和接收方比较熟悉的攻击者就可能尝试这种密钥。即使攻击者没有关于发方和收方详细的资料，一个聪明的攻击者也可能会尝试英语中 10 个字母的短语作为密钥。尽管全部尝试这些密钥的工作量会让人感到畏惧，但是英语中 10 个字母的短语的数量比起 10^{10} 还是要小得多。这种攻击方法我们通常称为**字典攻击**。

注意，如果明文消息和密钥都不是没有意义的文字时，字典攻击就有可能成功。无论如何，如果明文消息本身就是没有意义的，特别是它包含一些 ASCII 码中不可打印的字符时，攻击者就很难得到正确的解密。

另外一个使用 OTP 密码过程中的诱惑就是密钥的重复使用，这是一个致命的错误，因为这使得 Friedman 的重合指数攻击成为可能，就如同密码此时已成了维吉尼亚密码。但是如果不重复使用密钥，那么这样一个巨大的密钥的产生和分发就成了一个非常困难的问题，如同发送消息本身一样麻烦。这就是**密钥分配**和**密钥管理**基本问题的第一个例子。任何实际的加密讨论都会涉及到这个问题。

如果 OTP 密码被正确使用，即没有重复使用密钥，那么对已知明文的攻击就没有意义了，因为获得的密钥除了能得知当次的明文外不会再有别的用处。同样，选择明文攻击也就没有意义。

另外一个潜在的问题就是，当明文消息特别长时，编排同样长度且随机的密钥就成为了

问题。如果对这种很长明文消息的密钥不是很随机，则可能会给几种唯密文攻击提供条件，我们在后面讨论其他古典密码时还将会看到这一点。

术语：一次一密密码是**对称的**，即知道加密密钥就等于知道了解密密钥，反之亦然。另外它还是**多表代替**，在整个消息中同一个字符通常不会被加密成同一个字符。

习题

1.3.01 用密钥为“excelsior”的 OTP 密码加密“nevermore”。

1.3.02 用密钥为“idiosyncratic”的 OTP 密码加密“nevermore”。

1.3.03 用密钥为“tangentially”的 OTP 密码加密“nevermore”。

1.3.04 用密钥为“inconsequential”的 OTP 密码加密“nevermore”。

1.3.05 用密钥为“antedeluvian”的 OTP 密码加密“dog has fleas”。

1.3.06 用密钥为“hypeoversubstance”的 OTP 密码加密“idealism”。

1.3.07(*) 已知长度为 8 字符的 OTP 密钥为英语字母，而非任意 8 字符的字符串。估计一下寻找这个密钥的工作量（a 到 z 被编码成 0 到 25）。

1.4 仿射密码

至此我们已经知道，移位密码由于它的密钥空间太小，使它容易受到穷举唯密文攻击。（并且它对已知明文攻击和选择明文攻击都是透明的。）另一方面，尽管一次一密密码是完善保密的，但其密钥生成比较困难，密钥不能重复使用，密钥分配成为一个巨大的问题。因此我们还是回到移位密码讨论上，并试图对它做一些改进。

仍然使用将字母编码为数字，同时进行模 26 约简计算的思想，我们考虑如下**仿射密码**（the Affine Cipher）的加密函数，使移位密码的密钥空间稍微增大一些：

$$E_{a,b}(x) = (a \cdot x + b)\%26$$

这里的 a,b 为介于 0 ~ 25 之间的整数。这实际上是对移位密码的推广，即 $a=1$ 即为一般的移位密码。比如

$$E_{3,11}(\text{hello how are you}) = \text{GXSSB GBZ LKX FBT}$$

（我们仍然用小写字母表示明文，大写字母表示密文。）因此，这里的密钥是一对整数 (a,b)。它仍然是一个单表代替密码，只不过是密钥空间增大了一些罢了。当然这个密码也是对称密码，即加密密钥和解密密钥实质上是一回事。

当然，我们也希望密文只能以一种方式被解密。按照函数的语言，即我们需要 $E_{a,b}$ 有一个反函数，或者至少是一个左反函数。因此，我们希望加密函数 $E_{a,b}$ 是一个**双射**。实际上，解密函数和加密函数具有相同的形式。

命题 对于一个具有模 26 乘法逆 a^{-1} 的整数 a，仿射加密函数 $E_{a,b}$ 有一个反函数

$$D_{a,b} = E_{a,b}^{-1} = E_{a^{-1},-a^{-1}\cdot b}$$

证明 首先我们有 $E_{a,b} = E_{1,b} \circ E_{a,0}$，并且我们知道简单的移位密码 $E_{1,b}$ 有反函数 $E_{1,-b}$，所以如果 $E_{a,0}$ 有反函数就可保证 $E_{a,b}$ 有反函数。但是因为整数 a 具有模 26 乘法逆 a^{-1}，因此 $E_{a,0}$ 的反函数可由下式给出：

$$E_{a,0}^{-1}(x) = a^{-1} \cdot x\%26 = E_{a^{-1},0}(x)$$

这样利用反函数的复合，即 $(f\circ g)^{-1}=g^{-1}\circ f^{-1}$，可得

$$E_{a,b}^{-1}=(E_{1,b}\circ E_{a,0})^{-1}=E_{a,0}^{-1}\circ E_{1,b}^{-1}=E_{a^{-1},0}\circ E_{1,-b}=E_{a^{-1},-a^{-1}\cdot b}$$

这就证明了我们的结论。 ♣

如果我们认可这样一个结论，在 0 到 25 之间有 12 个数（除 13 以外的奇数）存在模 26 的逆，那么我们就是$12\times 26-1=311$个密钥(a,b)，而不是移位密码的 25 个密钥。（这是因为对 a 有 12 种选择，b 有 26 种选择，再去除 $(a,b)=(1,0)$）。至少从相对的意义上说，密钥空间比移位密码的有所增大。

因此，至少要实施**穷举**唯密文攻击（尝试 311 个可能的密钥，看那一个会产生有意义的输出），计算量也是有所增加的。

在关注针对更厉害的唯密文攻击方法进行更加有效的技术改进之前，我们先来看一下选择明文攻击和已知明文攻击，它们此时也会比使用移位密码时要相对难些。

我们先来考查选择明文攻击，因为这是最容易的：两个有选择的明文字符就足以揭示密钥。假设 (a,b) 是一个密钥，我们将选择两个不同的明文字符 x,y，这样就可由 $E_{a,b}(x)$ 和 $E_{a,b}(y)$ 的信息来确定密钥 (a,b)。选取 x=0（对应于字母 a），则

$$E_{a,b}(0)=(a\cdot 0+b)\%26=b\%26=b$$

因此密钥的一部分 b 即可由选择的一个明文字符直接得到。接着，由

$$E_{a,b}(1)=(a\cdot 1+b)\%26=(a+b)\%26$$

可进一步得到 a。这是因为 b 是已知的，用减法和模 26 约简即可。

已知明文攻击比选择明文攻击稍微复杂一些。具体过程如下，假设我们得到了两个在 0 到 25 间不同 x,y 的密文 $E_{a,b}(x)$ 和 $E_{a,b}(y)$，然而我们还不能控制 x,y 中的一个。那么

$$E_{a,b}(x)-E_{a,b}(y)=(a\cdot(x-y))\%26$$

由于不存在 b，因而我们可减轻计算难度。如果我们的运气够好的话，即如果 $(x-y)$ 存在一个模 26 的逆 $(x-y)^{-1}$，则

$$a=(x-y)^{-1}(E_{a,b}(x)-E_{a,b}(y))\%26$$

那么 b 就容易通过如下减法计算得到：

$$b=(E_{a,b}(x)-a\cdot x)\%26$$

因此，如果运气稍微好一点，若找到模 26 逆，有两个明文字符的已知明文攻击就会取得成功。

我们暂时先把唯密文攻击的讨论放一下，来看一看重复应用仿射密码会得到什么样结果，密钥可以相同也可不同。一般来说，如果一个加密方案 S 被应用一次，则称为**单轮** S。如果加密方案 S 被应用三次，则称为**三轮** S，经以此类推。重复应用一个加密方案的动机是这样的，如果单次应用能够“混淆”明文消息，那么多次应用后将使得明文消息“混淆”得更好。一些好的现代密码体制也是这样做的。

但是，许多古典密码的多轮应用并不能提供额外的安全性。这可由移位密码的多轮应用看出来：重复移位的效果等同于一次移位。对仿射密码可能还不太明显，但可由如下公式得出：

$$E_{a,b}\circ E_{c,d}=E_{ac,ad+b}$$

这就是说分别应用了密钥 (a,b) 和 (c,d) 的仿射密码与使用单个密钥 $(ac,ad+b)$ 的效果是一样的。继续这种重复应用的方式，则可以得到这样的结论：任何次数的仿射密码的重复应

用与一次这样的加密获得相同的效果。

多轮仿射密码（即使用不同的密钥）不会产生任何新的结果，这一事实也可用来说明这些密码不能很好地“混淆”0～25这个集合。

稍后我们还会利用英语的基本统计性质，给出一个对仿射密码的唯密文攻击。

利用Friedman的重合指数攻击（我们后面会用到维吉尼亚密码上），可以对仿射密码发起一个更具毁灭性的唯密文攻击。假设消息是连贯的英语，并且长度适当，则这种攻击可实现完全自动的有敌意的解密。当这种攻击完全自动时，即便适当增加密钥空间也是没有意义的。

习题

1.4.01 用仿射密码的加密函数 $E_{3,7}(x)=(3\cdot x+7)\%26$ 加密“meet me at midnight”。

1.4.02 用仿射密码的加密函数 $E_{11,1}(x)=(11\cdot x+1)\%26$ 加密“meet me at midnight”。

1.4.03 用仿射密码的加密函数 $E_{9,3}(x)=(9\cdot x+3)\%26$ 加密“meet me at midnight”。

1.4.04 用仿射密码的加密函数 $E_{11,19}(x)=(11\cdot x+19)\%26$ 加密“meet me at midnight”。

1.4.05 用仿射密码的加密函数 $E_{19,8}(x)=(19\cdot x+8)\%26$ 加密“meet me at midnight”。

1.4.06 确定加密函数为 $E_{3,7}(x)=(3\cdot x+7)\%26$ 的仿射密码的解密密钥。

1.4.07 确定加密函数为 $E_{11,5}(x)=(11\cdot x+5)\%26$ 的仿射密码的解密密钥。

1.4.08 确定加密函数为 $E_{7,3}(x)=(7\cdot x+3)\%26$ 的仿射密码的解密密钥。

1.4.09 确定加密函数为 $E_{13,16}(x)=(13\cdot x+16)\%26$ 的仿射密码的解密密钥。

1.4.10 给定一个正整数 m，并设 $E_{a,b}$ 是由 $E_{a,b}(x)=(ax+b)\%m$ 所定义的由集合 $\{0,1,2,\cdots,m-1\}$ 到自身的一个函数，证明 $E_{a,b}=E_{1,b}\cdot E_{a,0}$。

1.4.11 条件同1.4.10，证明 $E_{a,b}\circ E_{c,d}=E_{ac,ad+b}$。

1.4.12 已知明文攻击：假设 $E_{a,b}(3)=5$ 且 $E_{a,b}(4)=7$，求密钥 (a,b)。

1.4.13 已知明文攻击：假设 $E_{a,b}(3)=19$ 且 $E_{a,b}(4)=2$，求密钥 (a,b)。

1.4.14 已知明文攻击：假设 $E_{a,b}(3)=5$ 且 $E_{a,b}(6)=7$，求密钥 (a,b)。

1.4.15 已知明文攻击：假设 $E_{a,b}(11)=8$ 且 $E_{a,b}(4)=11$，求密钥 (a,b)。

1.4.16 已知明文攻击：假设 $E_{a,b}(8)=16$ 且 $E_{a,b}(19)=21$，求密钥 (a,b)。

1.4.17 已知明文攻击：假设 $E_{a,b}(8)=11$ 且 $E_{a,b}(11)=1$，求密钥 (a,b)。

1.4.18 已知明文攻击：假设 $E_{a,b}(9)=16$ 且 $E_{a,b}(20)=11$，求密钥 (a,b)。

1.4.19 已知明文攻击：假设 $E_{a,b}(3)=5$ 且 $E_{a,b}(5)=7$，求 a 和 b 的概率。

1.4.20 证明两轮仿射密码实际上不产生任何新的效果，并证明对任何字符 x（0到25），都有 $E_{25,25}(E_{25,25}(x))=x$。

1.4.21 证明三轮仿射密码可能实际上不产生任何新的效果，并证明对任何字符 x（0到25），都有 $E_{3,2}(E_{3,2}(E_{3,2}(x)))=x$。

第2章 概　率

本章的主要目标是：为探讨英文或其他自然语言的一些特征而介绍有关概率的简单概念。下面这个问题可以很好地阐明这个思想：当我们看见一串字母（数字和标点符号），我们将如何辨别“**它是不是英文**？”如果说一个会说英语的人能够辨别它是英文，这种回答太过于含糊其词，因为并没有告诉我们如何来辨别它是英文，以便于其他不会英文的人也能够识别它，或者把它描述为机器指令。对于这个问题我们并不准备在此给出一个更好的答案，只是讨论一些可被识别并以系统化机器方式加以应用的英文的某些低级特征。

最重要也是最基本的特征就是：单个字母出现的频率。这已经成为对弱密码进行攻击的重要内容，也可以让我们理解概率攻击的原理。

下面我们介绍一些概率的基本知识。首先介绍我们在概率统计中经常用到的一些计数原理，这些原则也将在本书后续内容中反复出现。

2.1 计数

这里有一些关于**计数**方面的例子。虽然简单，但是非常重要。这也是其他内容的预备知识。当然，我们这里讲的“计数”是指结构化的计数。

例 1　假如有 n 件不同的事物，如1到 n 之间不重复的整数，试问一共有多少种不同的**排序方式**？

$$i_1, i_2, i_3, \cdots, i_{n-1}, i_n$$

答案很显然，第一件事物 i_1 共有 n 种选择，第二件事物就只有 $n-1$ 种可能选择（因为我们不能重复选择已经用过的数），那么就 i_3 只剩下 $n-2$ 种选择（因为在任何情况下我们都不可以再选择 i_1 和 i_2 了！），以此类推，i_{n-1} 有 2 种可能，i_n 就只有一种可能了。因此，n 种不同的事物一共有：

$$n \cdot (n-1) \cdot (n-2) \cdot \cdots \cdot 2 \cdot 1$$

种可能的**排序**了。因为这类情况频繁出现，我们称之为 n 的**阶乘**，记为 $n!$，即有：

$$n! = n \cdot (n-1) \cdot (n-2) \cdot \cdots \cdot 2 \cdot 1$$

例 2　n 个元素的集合共有多少个包含 k 个元素的子集？第一次选择时共有 n 种可能，第二次剩下 $n-1$ 种可能（因为有一个元素已经移走了），第三次剩下 $n-2$ 种（因为在前二次操作后，有两个元素不能再选），以此类推，第 k 次就剩下 $n-(k-1)$ 种选择，即共有 $\dfrac{n!}{(n-k)!}$ 种。但这并不就是我们所求的结果，因为它包含的是所有不同排列的可能情况，但子集中的元素是不论顺序的，即：

$$\frac{n!}{(n-k)!} = k! \times \text{实际的结果}$$

这是因为由例 1 的结论可知，k 个事件共有 $k!$ 种排列，因此 n 个元素的集合中 k 个元素的子集数总共就有：

$$\frac{n!}{k!(n-k)!}=\binom{n}{k}$$

种可能。

$\frac{n!}{k!(n-k)!}$经常出现，所以需要给它一个名字和记号，因此它被称为**二项式系数**，记为

$$\frac{n!}{k!(n-k)!}=\binom{n}{k}$$

读作“n选k”。

例3 在一个有10个元素的集合中，有多少个3元子集和5元子集的不相交子集对（两集合不相交）？在第一次进行子集元素选择时有$\binom{10}{3}$种可能，而原始集合中的元素个数就只剩下10−3=7个，因此，为5元第二个子集选择元素时就只有$\binom{7}{5}$种可能。所以10个元素的集合中的3元和5元不相交子集对共有：

$$\binom{10}{3}\binom{7}{5}=\frac{10!}{7!3!}\frac{7!}{5!2!}=\frac{10!}{3!5!2!}$$

例4 在n元集合中，一共有多少个k元子集对且两两不相交？其中$2k\leqslant n$。首先，第一次选择k元子集时共有$\binom{n}{k}$种可能，原始集合中的元素个数剩下$n-k$个，于是第二次选择时共有$\binom{n-k}{k}$种可能。但是我们在计数需要考虑第一子集和第二子集顺序，事实上问题并非如此。现在我们知道两个事件（如子集）在排序时有2！=2种可能，因此在n元集合中的两两不相交的k元子集对的个数共有：

$$\frac{1}{2}\binom{n}{k}\binom{n-k}{k}=\frac{1}{2}\frac{n!}{(n-k)!k!}\frac{(n-k)!}{k!(n-2k)!}$$

$$=\frac{n!}{2k!k!(n-2k)!}$$

总结前面例子：对于整数n,l,k且$n\geqslant kl$，在一个n元集合中，l个不相交的k元子集族的个数是多少呢？我们有：

$$\binom{n}{k}$$

选择第一个子集的元素的可能有：

$$\binom{n-k}{k}$$

选择第二个子集的元素的可能有：

$$\binom{n-2k}{k}$$

选择第三个子集到第l个子集的元素的可能有：

$$\binom{n-(l-1)k}{k}$$

但是由于在计算过程中考虑了子集排序，所以我们在这里必须除以 $l!$ 才能得到集合族的实际值，可以得到最终准确结果：

$$\frac{n!}{l!(k!)^l(n-lk)!}$$

习题

2.1.01 集合 {1,2,3} 中共有多少种不同的排列方式？

2.1.02 集合 {1,2,3,4} 中共有多少种不同的排列方式？

2.1.03 集合 {1,2,3,4,5} 中共有多少种不同的排列方式？

2.1.04 集合 {a,b,c,d,e} 中共有多少种不同的排列方式？

2.1.05 从 1,2,3,…,6,7 中任选 3 个数的组合数是多少？

2.1.06 从 1,2,3,…,9 中任选 3 个数的组合数是多少？

2.1.07 集合 {1,2,3,4,5,6,7} 中有多少 4 元子集？

2.1.08 集合 {1,2,3,…,10} 中有多少 5 元子集？

2.1.09 集合 {1,2,3,…,12} 中有多少 4 元子集？

2.1.10 从集合 {1,2,…,10} 中选择不同的元素可组成多少个无序元素对？有序元素对又是多少？

2.1.11 n 元集合 S 中，不同大小的子集共有多少个？（提示：S 中的元素要么在子集中，要么不在子集中，所以对每个元素有两种选择。）

2.1.12 集合 {1,2,3,4,5,6,7,8} 中有多少 3 元不相交的子集对？

2.2 基本思想

尽管概率的基本思想在某些情形下并不与事实相符，但它还是我们的文化中的重要部分。正是由于这个原因，一些概率基础的合理结论与直觉并不相符，这就需要经过反复的思考，方能改变直觉并与之相符。

事实上，概率的数学化形式与真实世界的描述之间不相符的情形长期存在，以致于直到 1930 年以后概率才正式成为数学系统中的合法组成部分。我们习惯地认为随机性或者随机选择是有准确的含义，然而在现实生活中，它们意味着不清楚。譬如我们会认为十进制表示的 π 是“随机的”，然而我们并不能直接看到描述 π 的实际模式。

然而，对“概率”和“随机性”的抽象描述最起码能够对现实世界中的观点提供参考和启发式的定位。在本节的最后，我们将对**生日攻击**中计算的重要应用作全面的介绍。

首先，对待事件发生的几率，有一个从通俗观念到概率观念的转变过程。通常来说，某个事件发生的几率是用一个百分比来表示的，**概率**的取值是在 0 到 1 之间的某个数值。最简单的转换规则如下：如果说某个事件发生的几率为 $x\%$，那么从概率的角度来讲，它发生的概率为 $x/100$。（这种变换并没有引入新的内容。）

我们研究的第一个例子就是掷硬币问题。假设一枚“公平的”硬币它总是以“相等的几率”正面朝上或者背面朝上。而且，每次投掷的结果独立于其他的投掷。也就是说，某次投

掷的结果不能影响任何其他投掷的结果。那么（举例）10 次硬币投掷的结果应该是一半正面朝上、一半背面朝上。很少有全部 10 次投掷都是“正面”朝上的结果出现。迄今为止，通过这种含糊其词的介绍，我们对问题的本身还没有完全的认识。

但是，若以每 10 次投掷为一次试验，反复进行这样的试验，结果表明只有大概 1 / 4 次试验正好出现 5 次正面 5 次背面。（而大约进行 $2^{10}=1024$ 次试验中才有一次全部是正面朝上的结果。）事实上，大约有 2 / 5 的结果是 6 次正面与 4 次背面，反之亦然。也就是说，6—4 分布或 4—6 分布的结果比“预期”的 5—5 分布的结果更有可能出现。

但是这并不是自相矛盾，因为直觉告诉我们并不是正好一半正面一半背面，而只是大约一半对一半。下面我们将通过一种更好的方式来解释这一问题。

记录一个 n 次投掷的过程，每一枚硬币都有两个可能出现的结果，因此，n 次投掷结果的序列就有：

$$\underbrace{2\times\cdots\times 2}_{n}=2^n$$

种可能。由前面的假设条件可知硬币是“公平的”（fair）并且单独的硬币投掷并不影响其他投掷的结果，这就说明 2^n 个投掷结果序列中每一个序列都是等概率出现的。因此，任何单个序列出现的概率是所有 n 次可能结果的 $1/2^n$。而且，对于所有可能的 2^n 个结果序列组成的集合 A 及其子集 S，我们可以得出如下结论：

$$\text{给定一个结果属于 } S \text{ 的 } n \text{ 次投掷序列的概率}$$
$$=\frac{\text{集合 } S \text{ 中元素的个数}}{\text{集合 } A \text{ 中元素的个数}}=\frac{\text{集合 } S \text{ 中元素的个数}}{2^n}$$

那么 n 次投掷恰好有 k 次正面朝上的概率（$0\leqslant k\leqslant n$）计算如下：

$$n \text{ 次投掷中恰好有 } k \text{ 次正面朝上的概率}$$
$$=\frac{n \text{ 次投掷恰好有 } k \text{ 次正面朝上序列的个数}}{n \text{ 次投掷序列的总个数}}$$
$$=\frac{n \text{ 次投掷恰好有 } k \text{ 次正面朝上序列的个数}}{2^n}$$

为了计算 n 次投掷恰好有 k 次正面朝上的概率，我们可以将这个问题转化为在 n 元集合中求 k 元子集的问题。也就是说，全集就是 n 次硬币投掷所有可能的结果组成的集合，子集就是我们要求解的正面朝上的结果组成的集合。数值上就等于二项式系数：

$$n\text{选}k=\frac{n!}{k!(n-k)!}=\binom{n}{k}$$

举例来说，10 次投掷的结果中恰好有 5 次正面朝上的概率就是：

$$\frac{\binom{10}{5}}{2^{10}}=\frac{\left(\frac{10\cdot 9\cdot 8\cdot 7\cdot 6}{5\cdot 4\cdot 3\cdot 2\cdot 1}\right)}{1024}=\frac{252}{1024}\approx\frac{1}{4}$$

正如前面所分析的结果一样。而 6—4 组合或者 4—6 组合情况出现的概率就是：

$$\frac{10 \text{ 次投掷中恰有 } 6 \text{ 次正面或 } 4 \text{ 次正面的序列的个数}}{2^{10}}$$

$$=\frac{\binom{10}{4}+\binom{10}{6}}{1024}=\frac{2\cdot(10\cdot9\cdot8\cdot7/4\cdot3\cdot2)}{1024}=\frac{420}{1024}\approx\frac{2}{5}$$

这样的结果并不令人吃惊，在$2n$次投掷中，正好得到一半正面一半背面结果的概率将随着投掷次数的增加而下降。实际上，投掷的次数趋于无限大时，这个值将趋向于0：

$$\frac{\binom{2}{1}}{2^2}\approx0.5$$

$$\frac{\binom{4}{2}}{2^4}\approx0.375$$

$$\frac{\binom{6}{3}}{2^6}\approx0.3125$$

$$\frac{\binom{8}{4}}{2^8}\approx0.2734$$

$$\frac{\binom{10}{5}}{2^{10}}\approx0.2461$$

$$\frac{\binom{12}{6}}{2^{12}}\approx0.1813$$

$$\frac{\binom{14}{7}}{2^{14}}\approx0.1683$$

很显然不会正好出现一半正面一半背面，而只是接近于这个值。这意味着什么呢？过去我们所用的方法，并不是一种十分确切的方法，而只是利用了**极限频率**的主要思想。极限频率的定义如下：设n次投掷得到正面朝上结果的次数用$h(n)$来表示，那么随着投掷次数n的不断增加，比率$h(n)/n$将逐渐接近1/2。或者，用极限的语言表示就是：

$$\lim_{n\to\infty}\frac{h(n)}{n}=\frac{1}{2}$$

我们称正面朝上的**极限频率**为1/2。对于任何无限次数的投掷，这个结果都应是1/2。

另外一种比较复杂的例子就是从缸中选彩色球问题。假设缸中有N个球，其中r个红色球和$b=N-r$个蓝色球，球的材料、重量和大小均一致。如果由一名被蒙住眼睛的人来从一个装有10个球的缸任选一个，那么很显然，选出红色球的概率是$r/10$，选出蓝色球的概率是$b/10$。反复进行这一试验，每次取一球，记下球的颜色（当然是暂时不蒙住眼睛），然后将球放入缸中，再进行下一次选球。在试验当中，每次取球不影响下一次的取球任务。因此它们之间是相互**独立的**，经过n次试验得到红球的结果记为$r(n)$。那么，与前面的讨论类似，我们可以得到任何无限次数试验后得到红球的极限频率为：

$$\lim_{n\to\infty}\frac{r(n)}{n}=\frac{r}{N}$$

从反面来考虑这个问题：如果缸中共有 N 个球，一些红色球和一些蓝色球，如果 $r(n)$ 表示 n 次试验中取得红球的次数，并且有（试验次数趋于无限）

$$\lim_{n\to\infty}\frac{r(n)}{n}=f$$

那么我们就可以推断出：

$$\text{缸中红色球的总数}=f\cdot N$$

也就是说，我们可以从取出红色球的极限频率 f，推断出某一次单独试验取出红色球的概率也是 f。

更进一步来考虑，一个缸中有 N 个多种颜色的球（除颜色外，球的其他属性完全一样）。对于任一颜色 c，如果缸中共有 n_c 个颜色为 c 的球，那么我们可以假定（在一次试验中）从缸中取出颜色为 c 的球的概率为 n_c/N。如果将 n 次试验取得 c 颜色球的次数记为 $n(c)$，类似地我们就可以推断出**极限频率** $\lim\frac{n(c)}{n}$ 的值为：

$$\lim_{n\to\infty}\frac{n(c)}{n}=\frac{n_c}{N}$$

反过来，如果已知取出颜色为 c 的球的极限频率为 f，那么就可以认为缸中那种颜色球的总数为 $f\cdot N$，并且我们认为取出 c 颜色球的概率为 f。

其实我们反复用“频率”来描述概率会有严重的缺陷，但是在没有给出严格的定义之前，这使得我们更容易理解概率的朴素思想，而且这种描述已经给我们了解这种思想开了个好头。我们将逐步抛开这种通俗的观点。

假如做某个试验 X，所得的不同结果分别为 $x_1,x_2,\cdots,x_n$。收集所有可能的结果组成一个**样本空间**。每个执行试验 X 的过程称为一次**试验**，每种可能的结果 x_i 称为一个**事件**。我们假设所有的结果 x_i 发生的**概率**为 $p_i\geqslant 0$，并有：

$$p_1+p_2+\cdots+p_n=1$$

用标准的符号表示则为：

$$P(X=x_i)=P(x_i)=p_i$$

我们把它读作：x_i 的概率为 p_i，或者是 $X=x_i$ 的概率是 p_i。

事件（有时也称为**复合事件**）的一个更为一般的概念就是样本空间的任何子集 A，样本空间可表示为 $\Omega=\{x_1,x_2,\cdots,x_n\}$，它就是所有最小事件的集合。$A$ 的**概率**就是：

$$P(A)=\sum_{x_i\in A}P(x_i)$$

即对集合 A 中的所有点 x_i 的概率进行求和。事件 A **发生**，如果 $x_i\in A$ 发生。因此，对于 $A=\{x_{i1},\cdots,x_{ik}\}$ 则有：

$$P(A)=P(x_{i1}\text{或}x_{i2}\text{或}\cdots\text{或}x_{ik})$$

作为极端情况：$P(\Omega)=1$，$P(\phi)=0$。

一般地，对于任何事件 A，非 A 事件可以用事件 A 的补集 $A^c=\Omega-A$ 来表示。于是

$$P(\text{非}A)=P(A^c)=P(\Omega-A)=1-P(A)$$

对于两个事件 A 和 B，事件 A **或者** B 可以用 $A\cup B$ 来表示，并且有：

$$P(A或B)=P(A\cup B)$$

对于两个事件 A 和 B，事件 A **并且** B 可以用 $A\cap B$ 来表示，而且有：

$$P(A且B)=P(A\cap B)$$

两个事件 A 和 B 是**不相交的**或者是**互相排斥的**，如果 $A\cap B=\varnothing$。这样我们就有

$$P(A或B)=P(A\cup B)=P(A)+P(B)$$

我们可以通过直觉知识、极限频率或者其他方法来对概率 p_i 进行赋值。事实上，它们可以通过实验的方式来测量，也可以通过某些启发式的方式来赋值。例如在上面讲述的极限频率的例子当中，我们可以反复地进行试验 X，后面试验的结果并不影响前面的结果，也就是说，它们是**相互独立的试验**。在 n 次这种独立试验中出现 x_i 的次数记为 $n(x_i)$。假设进行无限多次试验，并且 $p_i=\lim\limits_{n\to\infty}\dfrac{n(x_i)}{n}$ 值存在。那么**极限频率** p_i 就是事件 x_i 发生的**概率**。

例如，考虑从缸中取球的问题，假如缸中有 3 个红色球、3 个蓝色球和 4 个白色球（除颜色外，球的其他属性均相同）。由直觉知道取出任一球的概率为 1/10。（由于我们每次只取出一个球，所以“原子”事件实际上是互相排斥的）因此，最小事件 $x_1,x_2,\cdots,x_{10}$ 就表示从 10 个球中每次选出一个球。由于在每次取球时具有等同的几率，所以概率 $p_i=P(x_i)$ 就完全相等(并且和为 1)：

$$p_1=p_2=p_3=\cdots=p_{10}$$

如果（复合）事件 A 表示“取出的是红色球”，那么 A 就是一个三元子集，包括三个元素：“取出第一个红色球”、“取出第二个红色球”和“取出第三个红色球”。因此

$$P(A)=\frac{1}{10}+\frac{1}{10}+\frac{1}{10}=\frac{3}{10}$$

如果事件 B 表示“取出一个白色球”，由于事件 A 与事件 B 相互独立，所以取到要么是红色球要么是白色球的概率就是两者之和：

$$P(A\cup B)=P(A)+P(B)=\frac{3}{10}+\frac{4}{10}=\frac{7}{10}$$

更进一步，我们进行两个不同的试验 X 和 Y，分别得到的可能结果为 $x_1,x_2,\cdots,x_m$ 和 $y_1,y_2,\cdots,y_n$。如果对于所有的下标 $1\leqslant i\leqslant m$ 并且 $1\leqslant j\leqslant n$ 有：

$$P(X=x_i且Y=y_j)=P(X=x_i)\cdot P(Y=y_j)$$

那么就称 X 和 Y **相互独立**。

作为一个基本运算的例子，我们有如下的结果。这个命题的结论在今后的概率讨论中会经常用到。

命题　在试验 X 中，不同结果 $x_1,x_2,\cdots,x_n$ 的概率分别为 $P(x_1)=p_1,\cdots,P(x_n)=p_n$。集合 A 为样本空间 $\{x_1,\cdots,x_n\}$ 的一个子集，并有 $P(A)=p$，设整数 $k\leqslant N$ 且 $k\geqslant 0$，$N>0$。那么 N 次试验中集合 A 恰好出现 k 次的概率为：

$$\binom{N}{k}\cdot p^k(1-p)^{N-k}$$

证明　当 $N=1$ 时，A 出现的概率为 p，二项式系数 $\binom{1}{1}$ 为 1。集合 A 不出现的概率为 $1-p$，

二项式系数$\binom{1}{0}$也为1。

对N进行归纳：因为不同的试验之间是相互独立的，所以有：

$$\begin{aligned}&P(A\text{在}N\text{次试验中出现}\ \ k\text{次})\\&=P(A\text{在前}\ N-1\text{次试验中出现}\ k\text{次})\cdot P(A\text{在第}\ N\text{次试验中没有出现}\ )\\&+P(A\text{在前}\ N-1\text{次试验中出现}\ \ k-1\text{次})\cdot P(A\text{在第}\ N\text{次试验中出现}\ )\\&=\binom{N-1}{k}p^k(1-p)^{N-1-k}\times(1-p)\\&+\binom{N-1}{k-1}p^{k-1}(1-p)^{N-1-(k-1)}\times p\end{aligned}$$

现在我们要证明这个式子等于：$\binom{N}{k}p^k(1-p)^{N-k}$

比较两个式子可以看出，p和$1-p$的幂次已经匹配，因此我们只需证明：

$$\binom{N-1}{k}+\binom{N-1}{k-1}=\binom{N}{k}$$

展开二项式，等式左边就是：

$$\frac{(N-1)!}{k!(N-1-k)!}+\frac{(N-1)!}{(k-1)!(N-1-(k-1))!}$$

$$=\frac{(N-1)!\cdot(N-1-(k-1))}{k!(N-1-k)!\cdot(N-1-(k-1))}+\frac{(N-1)!\cdot k}{k\cdot(k-1)!\ (N-1-(k-1))!}$$

$$=\frac{(N-1)!\cdot[(N-1-(k-1))+k]}{k!(N-k)!}=\frac{(N-1)!\cdot N}{k!(N-k)!}=\frac{N!}{k!(N-k)!}=\binom{N}{k}$$

命题证明完毕。 ♣

如果将所有可能结果组成的集合记为Ω，（复合）事件A发生的概率$P(A)>0$，B为另一（复合）事件。那么**条件概率**（假定A出现时B出现的概率）就可以用符号$P(B\mid A)$来表示，其计算公式为：

$$P(B\mid A)=\frac{P(A\cap B)}{P(A)}$$

公式$P(B\mid A)=P(A\cap B)/P(A)$表明，我们可以通过计算另外两个概率来计算条件概率，而在实际情况下，由于某种原因，也许我们可以直接得到$P(B\mid A)$。如果我们知道$P(A)$的话，就有公式：

$$P(A\text{且}B)=P(A\cap B)=P(B\mid A)\cdot P(A)$$

这个结果将在下面的例子中引用。

下面我们将通过另外一种重要的基本计算来解释**生日攻击**(birthday attacks)的工作原理。令

$$\Omega=\{1,2,\cdots,N\}$$

为所有“原子”事件的样本空间。假定每个原子事件具有相同的概率，即对于所有 $i=1,2,...,N$ 都有：

$$P(i)=\frac{1}{N}$$

试验 X 就是从样本空间 Ω 中随机选择一个元素（并不是从集合中移走元素）。问题就是：n 次试验后至少 2 次结果相同的概率是多少？

命题　在上述条件下，n 次试验后至少 2 次结果相同的概率至少为

$$1-e^{-\frac{1}{2}(n-1)n/N}$$

因此，对于

$$n>\sqrt{2\ln 2}\sqrt{N}$$

出现两次相同结果的概率至少为 1/2。

证明　首先计算出没有两种完全相同结果出现的概率，然后用 1 减去它就可以得到我们期望的结果。我们假定 n 次试验结果有序排列，并根据前 $n-1$ 次试验的结果来计算出 n 次试验中没有两种相同结果的概率。

经过 1 次试验，只有一种结果，所以没有两种结果相同的概率为 1。

经过 2 次试验，只有 $1/N$ 的机会使得第二次结果与第一次结果相同（无论第一次的是何种结果）。因此，两次结果不相同的概率就是 $1-\frac{1}{N}$。

经过 3 次试验，如果假定前两次的结果不同，则在此条件下第三次试验结果与前两次结果中的一个相同的条件概率为 $2/N$。因此，给定前两次结果不同，第三次的结果也与前两次的结果都不相同的条件概率就为 $1-\frac{2}{N}$。因为前两次结果不相同的概率就是 $1-\frac{1}{N}$，所以：

$$P(\text{前3次结果均不同})=(1-\frac{1}{N})(1-\frac{2}{N})$$

经过 4 次试验，假设前三次试验结果不相同，第四次结果与前三次中任一种结果相同的条件概率为 $3/N$。因此，在前三次试验结果不相同的假定情况下，第四次与前三次结果都不相同的条件概率为 $1-\frac{3}{N}$。利用前面的结果可得：

$$P(\text{前4次结果均不同})=(1-\frac{1}{N})(1-\frac{2}{N})(1-\frac{3}{N})$$

以此类推：我们可以得出：

$$P(n\text{次试验结果均不相同})=(1-\frac{1}{N})(1-\frac{2}{N})(1-\frac{3}{N})\cdots(1-\frac{n-1}{N})$$

所有结果均不相同的概率的对数表示为：

$$\ln(1-\frac{1}{N})+\ln(1-\frac{2}{N})+\cdots+\ln(1-\frac{n-1}{N})$$

当然我们假定 $n<N$，否则整个过程就没有意义了。然后调用函数 $\ln(1-x)$ 当 $|x|<1$ 时的一阶泰勒级数展开式：

$$\ln(1-x)=-(x+\frac{x^2}{2}+\frac{x^3}{3}+\frac{x^4}{4}+\cdots)$$

特别地，对于$0<x<1$，下面的表达式成立：

$$\ln(1-x)\leqslant -x$$

因此，

$$\ln(1-\frac{1}{N})+\ln(1-\frac{2}{N})+\cdots+\ln(1-\frac{n-1}{N})\leqslant -(\frac{1}{N}+\frac{2}{N}+\cdots+\frac{n-1}{N})$$

我们也许还记得公式：

$$1+2+3+4+\cdots+(k-1)+k=\frac{1}{2}k(k+1)$$

运用这个公式，我们就可以估计：

$$\ln(P(n\text{次试验结果均不相同}))\leqslant \frac{-\frac{1}{2}(n-1)n}{N}$$

由于指数函数是递增函数，所以上式两边作为函数e^x的指数，即可得到n次试验结果均不相同的概率$P(n\text{次试验结果均不相同})$的估计值，然后用1去减去这个估计值，就可以得出命题给出的结论。

如果n越来越大，表达式$n(n-1)$与n^2的实际差距越来越小，因此我们可得近似的公式：

$$\ln(P(n\text{次试验结果均不相同}))\leqslant -\frac{n^2}{2N}$$

所以当没有两种相同结果的概率小于1/2时，至少出现两次相同结果的概率就会大于或等于1/2。因此，给定一个N，我们就可以找到一个满足条件的最小的n使得：

$$-\frac{n^2}{2N}<\ln\frac{1}{2}$$

由此我们便可得出命题的结论。 ♣

习题

2.2.01 在10次掷硬币试验中，恰有3次正面朝上的概率是多少？

2.2.02 在10次掷硬币试验中，恰有5次正面朝上的概率是多少？

2.2.03 在6次掷硬币试验中，正面出现的次数严格地多于背面出现次数的概率是多少？

2.2.04 在8次掷硬币试验中，正面出现的次数严格地多于背面出现次数的概率是多少？

2.2.05 在9次掷硬币试验中，正面出现的次数严格地多于背面出现次数的概率是多少？

2.2.06 如果一个缸中有3个红色球和7个蓝色球，则在2次试验中取到2个红色球的概率是多少？

2.2.07 如果一个缸中有5个红色球和6个蓝色球，则在2次试验中取到2个红色球的概率是多少？

2.2.08 如果一个缸中有3个红色球和7个蓝色球，则在10次试验中至少4次取到红色球的概率是多少？

2.2.09 证明$1+2+3+4+\cdots+(n-1)+n=\frac{1}{2}n(n+1)$。

2.2.10 在一次掷两个骰子的试验中，得到“7点”或者“8点”的概率是多少？得到“2点”的概率又是多少？

2.2.11 在$3N$次掷硬币试验中，至多有N次正面朝上的概率是多少？

2.2.12 生日悖论：证明在23个人中间，至少有两个人生日相同的概率大于1/2。

2.2.13(*) 在本节最后一个命题的证明过程中，如果我们使用简单的估计$\ln(1-x)<-x$，对$0<x<1$，则会得到怎样的结果？

2.2.14(**) 假设随机选择0与1之间的两个实数，则它们的和大于1的概率是多少？它们的乘积大于1/2的概率又是多少？

2.3 英文统计

对于英语句子来说，它有什么样随机字符流不具有的特征呢？又怎样用纯机械式的术语来描述它呢？特别地，我们如何能够判断一个所谓破译的译码真的是报文呢，还只是乱语？

在某些情况下，当我们破译一条消息时，总是尝试猜测一个符合实际的消息，这个消息中已知有一些片语是符合解译规则的，通过这种方式以达到进一步的破解。

在这种情况下，只要假定消息不是乱语，就能够立即得到很容易辨认的正确报文。尤其是在计算机问世以前的古典密码分析学中，对“部分明文”内容的分析是一种最基础的技术。如果消息纯粹就是一些乱码，那么正确译文的内容与错误译文的内容是不可分辨的，整个破译的过程就是无效和没有意义的。

对于那些假定前后一致而且“有意义的”明文来说，有一个问题就是用一种自动过程来从不真实的信息中辨认出那些似是而非的信息。这些事实充分而准确地说明英文句子具有随机字符流不具有的特征。目前我们并不准备去讨论英文句子的意义，而只是研究它们的形式。甚至我们只是用比较低级的统计方法来研究这个问题。

因此，我们选取一些英文明文的特征来大致有两方面的目的：首先，隐蔽这些特征以防止敌方破坏我方的密码系统。其次，这些特征有助于我们破译敌方的密码系统。在这一部分我们只简单描述其中一些特征，而将利用和隐蔽这些特征的内容放在以后的部分讨论。

我们使用的主要技术是一种启发式技术，也就是利用字母或单词在英文经典样本中出现的**频率**的一种技术。例如经过简单的观察就可以发现字母“e”出现的次数比字母“z”要多得多。由此可以看出，我们假定英文文本就是根据概率选择而得出的字母序列，那么“概率”f_e就是字母“e”出现的近似的比率：

$$\frac{\text{字母“e”在整个文本中出现的次数}}{\text{整个文本的字符总数}}$$

类似地，“概率”f_{the}就是单词“the”在英文文本中出现的比率：

$$\frac{\text{“the”在整个文本中出现的次数}}{\text{整个文本的字符总数}}$$

因此，假定为了“计算”一个字符是“e”的概率，我们可以先统计在一个大的样本文本中字母“e”出现的次数，然后再除以整个样本的字符总数。为了“计算”一个单词是“the”的概率，我们可以先统计在一个大的样本文本中单词“the”出现的次数，然后再除以整个样本中的单词总数。

这种思想与概率的**极限频率**概念（前面介绍过）一致，它实际上是一种试探法，因为英文中字符和单词的选择并不是一个随机的过程。然而，在计算机出现之前，在密码分析中这种试探法被证明是非常有效的。很明显，某些字母出现的次数比其他字母出现的次数多得多，

尤其是某几个字母或短单词的组合。因此，即使从很简单的概率试探法观点出发，英文与字母的随机序列之间有显著的差别。这就是对古典密码体制实施唯密文攻击的重要依据。

英文的一些显著特征如下：

短单词（small words）。在英文中只有很少几个非常短的单词。因此，如果在一个加密的文本中可以确定单词的范围，那么就能得出明显的结果。一个字母的单词只有 a 和 I。如果不计单词的缩写，在从电子邮件中选取 500k 字节的样本中，只有两个字母的单词仅出现 35 次，而 2 个字母的所有组合为 26×26＝676 种。而且，还是在那个样本中，只有 3 个字母的单词出现 196 次，而 3 个字母的所有组合为 26×26×26＝17 576 种。

常用单词（common words）。再次分析 500k 字节的样本，总共有 5000 多个不同的单词出现。在这里，9 个最常用的单词出现的总次数占总单词数的 21%，20 个最常用的单词出现的总次数占总单词数的 30%，104 个最常用的单词占 50%，247 个最常用的单词占 60%。本节最后列出了一些常用的单词及其使用频率列表（以百分比的形式）。

空格（blanks）。在普通的英文中，对“A”到“Z”以及空格（忽略大小写）进行统计表明，空格是使用次数最多的字符：大约 17%~18%的字符是空格，而其次最常用的字符如“e”，“t”，“o”，“a”和“i”等出现的次数均小于 9%。因此，如果将空格也作为字符并加密的话，这种高使用频率会泄露信息。另一方面，如果不对它们加密，那么将更直接地泄露其他信息：密码分析者就可以利用短单词频率和字母频率统计信息。基于这些原因，在“古典”密码系统中常常去除消息中的空格。

字母频率（Character frequency）。在 1M 字节旧的电子邮件文本（去掉所有的数据头）中，对字母“A”到“Z”（忽略大小写）分别进行统计，我们发现近似频率（以百分比）

e	11.67	t	9.53	o	8.22	i	7.81	a	7.73	n	6.71	s	6.55
r	5.97	h	4.52	l	4.3	d	3.24	u	3.21	c	3.06	m	2.8
p	2.34	y	2.22	f	2.14	g	2.00	w	1.69	b	1.58	v	1.03
k	0.79	x	0.30	j	0.23	q	0.12	z	0.09				

也就是说，11.67%的字母为“e”，9.53%的字母为“o”等等，直到最后 0.09%的字母为“z”。特别地，字母“e”出现的次数是字母“z”出现次数的 100 多倍，而其他字母的出现频率则居于两者之间。因此，就存在一个十分显著的统计偏差，也就是说 26 个字母并不是以相同的频率出现，而是差别很大。

目前使用这类信息很容易被误导，因为即使是从过滤的邮件中选择 500 000 字符也不是英文的随机样本。而且更重要的是，有时需要了解某些与机率相背离的可能性。这种背离在一些小样本中很有可能发生。实际上有些人在撰写小说时并不使用字母“e”。

而且，在没有经过仔细检验的情况下，仅凭借单个字符的频率是不是与已有的统计结果吻合，而断定一个字符流是英文，这种判断的合理性到底怎么样还不是很明确。从另一方面来说，我们根本不清楚英文与非英文字符流的频率有什么区别。一种更严格、更合理的统计研究将考虑计数样本大小及其他因素。

双字母是两个相邻字母的组合。共有 26×26＝676 种不同的双字母，一个字符流中双字母的出现可以比单个字符的频率提供更多的信息。实际上，字母表上每个字母在许多单词中都会出现，而双字母则不一定。我们可以讨论双字母，包括空格在内的双字母等在单词界上出现情况。

同样在从电子邮件（去除数据头）中选取的大小为 500k 字节的样本中，保留其中的空格，在 676 种双字母组合中的只有 611 种双字母出现。（如果去除空格，那么在 676 种组合中的有 659 种双字母出现。）前 44 种双字母组合占总数的 50%以上，前 102 种占 75%，前 175 种占 90%，前 359 种占 98%。最常用的双字母组合表在本节最后部分给出。

三字母是指相邻的三个字符组合。共有 26×26×26＝17576 种可能的三字母组合，相对来讲，经常出现的三字母组合很少。在 1M 字节过滤后的邮件样本中，只观察英文单词中出现的三字母组，前 241 种组合就已占所有三字母组的 50%。极端情况是：不到总数 1/70 的三字母组占所有出现的三字母组的 50%！前 652 种组合占 75%，前 1271 种组合占 90%，而前 2520 种组合占所有出现的三字母组的 98%。如果去除空格，前 430 种组合占 50%，前 1162 种占 75%，前 2314 种占 90%，而前 4408 种占 98%。最常用的三字母组合表在本节最后部分给出。

这里有个需要注意的地方。首先，正如观测单个字符出现的频率一样，英文文本的字符流并不完全是随机的：有些字母永远比其他字母出现的次数要多得多。其次，有些相邻二元字符组合（双字母）出现的次数也比其他的要多得多。类似地，相邻三元字符组合（三字母）也一样。一方面，这种低级的统计偏差可以运用到密码分析中。另一方面，在安全的密码体制中就必须掩盖。

作为一个特定的统计特征，空格出现的频率是任何其他字符的二倍。既然在去除空格之后，消息还能够清楚地辨认，那么在很多场合，加密之前就应将空格去除。要不然这些被去除的信息就可能被密码分析者利用。例如，在进行双字母和三字母组合统计时表明，如果单词界限明确，统计偏差显著不同，这与去除单词界限的情况形成了明显的对比。

余下的问题就是如何系统（有效）地利用这些信息。后面将给出一些简单的例子。

样本中 100 个最常用的单词占单词总数的百分比为：

the	4.65	to	3.02	of	2.61	i	2.2	a	1.95
and	1.82	is	1.68	that	1.62	in	1.57	it	1.22
for	1.17	you	1.06	be	0.99	not	0.84	on	0.76
have	0.71	this	0.69	as	0.57	at	0.56	would	0.55
are	0.55	but	0.54	if	0.53	my	0.53	with	0.5
your	0.48	so	0.48	or	0.46	some	0.43	will	0.41
do	0.39	about	0.39	me	0.38	from	0.35	by	0.33
no	0.33	more	0.33	what	0.32	an	0.32	there	0.32
one	0.32	all	0.32	was	0.30	we	0.30	just	0.27
which	0.27	can	0.26	very	0.25	series	0.25	am	0.24
things	0.24	people	0.24	get	0.23	hi	0.23	time	0.22
think	0.22	course	0.22	etc	0.22	also	0.21	any	0.21
other	0.20	than	0.2	know	0.19	could	0.19	they	0.19
too	0.19	only	0.18	up	0.18	good	0.18	out	0.18
has	0.17	such	0.17	had	0.16	should	0.16	now	0.16
dont	0.16	like	0.15	its	0.15	want	0.15	well	0.15

here	0.14	might	0.14	who	0.14	may	0.14	then	0.14
make	0.14	thanks	0.14	much	0.13	thing	0.13	did	0.13
how	0.12	really	0.12	he	0.12	students	0.12	maybe	0.12
yours	0.12	see	0.12	been	0.12	were	0.12	rather	0.11
when	0.11	paper	0.11	even	0.11	our	0.11	still	0.11
case	0.11	since	0.11	while	0.11	use	0.1	ill	0.10
email	0.10	stuff	0.10	seems	0.10	them	0.10	book	0.10
work	0.10	please	0.10	online	0.10	into	0.10	does	0.10
two	0.10	university	0.09	little	0.09	page	0.09	number	0.09

注意：有些单词可能会达到列表中“高频率”段。例如，如果样本不是从大学教授的信件中得来的话，“university”，“number”和“students”这几个单词的频率很可能就是另一回事了。如果类似信件被过滤掉的话，这种偏差在科技词汇统计时更加明显。一方面，我们希望得到更为通用的统计结果。另一方面，任何有关敌方的特殊信息都有潜在的用途。

样本中前77种双字母组合在单词中出现的百分比：

th	3.18	in	2.59	he	2.17	er	1.95	re	1.85	on	1.63	an	1.59
at	1.54	ou	1.43	or	1.26	es	1.26	ha	1.24	to	1.22	te	1.21
is	1.18	ti	1.17	it	1.16	en	1.13	nt	1.09	ng	1.08	al	1.07
se	1.05	st	1.01	nd	0.98	le	0.91	ar	0.90	me	0.90	hi	0.86
ve	0.85	of	0.84	ed	0.78	co	0.74	as	0.73	ll	0.72	ne	0.70
om	0.70	ri	0.68	ic	0.67	ro	0.67	ea	0.66	et	0.64	ur	0.64
io	0.64	ra	0.62	li	0.62	no	0.62	so	0.62	be	0.61	de	0.59
ma	0.59	si	0.58	ly	0.54	ut	0.53	ot	0.53	pr	0.53	fo	0.53
yo	0.52	il	0.50	ca	0.50	pe	0.50	ch	0.49	ho	0.49	ul	0.47
ce	0.47	ta	0.45	di	0.45	rs	0.45	el	0.44	ge	0.44	us	0.44
ec	0.42	ss	0.42	ac	0.41	ct	0.41	em	0.41	wh	0.41	oo	0.40

使用频率最高的77种双字母组合(含有空格)：

e_	3.15	_t	2.55	th	2.11	s_	1.97	t_	1.93	_a	1.81	in	1.72
_i	1.69	he	1.44	er	1.29	d_	1.24	re	1.23	_s	1.18	n_	1.15
on	1.08	an	1.05	_o	1.04	y_	1.03	at	1.03	r_	0.99	ou	0.95
o_	0.92	_w	0.92	or	0.84	es	0.83	ha	0.83	to	0.81	te	0.80
is	0.79	ti	0.78	it	0.77	en	0.75	nt	0.72	ng	0.72	_c	0.71
al	0.71	se	0.70	_m	0.69	_b	0.67	st	0.67	_p	0.65	nd	0.65
_f	0.62	le	0.60	ar	0.60	me	0.60	f_	0.59	l_	0.59	g_	0.58
hi	0.57	ve	0.57	_h	0.56	of	0.55	ed	0.52	_d	0.51	co	0.49
as	0.48	ll	0.48	ne	0.47	om	0.46	i_	0.45	ri	0.45	a_	0.45
_n	0.44	Ic	0.44	ro	0.44	ea	0.44	et	0.42	ur	0.42	io	0.42
_r	0.42	_e	0.41	ra	0.41	li	0.41	no	0.41	so	0.41	be	0.41

前 77 种双字母组合(去除空格):

th	2.63	in	2.08	he	1.75	er	1.67	re	1.52	on	1.33	es	1.32
an	1.29	at	1.28	ti	1.26	nt	1.16	ou	1.16	to	1.13	st	1.12
ha	1.05	or	1.05	et	1.03	en	1.01	te	1.01	is	0.98	it	0.97
ea	0.93	se	0.90	al	0.89	ng	0.89	nd	0.81	ed	0.76	hi	0.75
le	0.75	ar	0.74	si	0.73	me	0.73	so	0.71	of	0.70	ve	0.68
ri	0.64	as	0.64	om	0.64	ra	0.61	no	0.61	ne	0.60	co	0.60
ro	0.59	ll	0.59	ta	0.58	ic	0.57	ot	0.57	tt	0.57	li	0.57
yo	0.52	ur	0.51	ec	0.51	io	0.51	de	0.51	di	0.51	ma	0.51
ei	0.49	be	0.49	sa	0.47	ss	0.47	el	0.46	em	0.46	rs	0.45
fo	0.44	ut	0.44	ly	0.44	rt	0.43	ca	0.42	pr	0.42	na	0.42
ts	0.41	ho	0.41	il	0.41	pe	0.40	ch	0.40	ul	0.38	ee	0.38

英文单词中最常用的 77 种三字母组合的百分比:

the	2.44	ing	1.26	and	0.82	hat	0.78	tha	0.77	ion	0.75	you	0.67
ent	0.66	for	0.63	tio	0.63	thi	0.60	her	0.51	ati	0.47	our	0.47
ere	0.45	all	0.43	ter	0.43	ver	0.40	not	0.40	hin	0.40	ome	0.36
oul	0.36	uld	0.36	int	0.34	rea	0.34	pro	0.34	res	0.33	ate	0.33
hav	0.30	ave	0.30	ill	0.30	his	0.30	com	0.30	ons	0.30	are	0.28
ple	0.28	ers	0.28	con	0.27	ess	0.27	out	0.27	one	0.26	ith	0.25
som	0.25	ive	0.25	tin	0.25	nce	0.24	ble	0.24	ted	0.24	han	0.23
ine	0.23	per	0.23	ect	0.23	nte	0.23	wit	0.22	men	0.22	but	0.22
wou	0.21	ica	0.21	eve	0.21	cal	0.21	pre	0.21	cou	0.21	lin	0.21
est	0.20	eri	0.20	mor	0.20	ser	0.20	ore	0.19	any	0.19	abl	0.19
tic	0.19	urs	0.19	ant	0.19	sti	0.18	ear	0.18	hou	0.18	ies	0.18

最常用的 77 种三字母组合(含空格):

_th	1.67	the	1.22	he_	0.80	ing	0.63	_to	0.62	to_	0.55	ng_	0.52
_an	0.50	_in	0.49	_of	0.49	at_	0.45	is_	0.44	of_	0.44	e_t	0.43
on_	0.43	er_	0.42	nd_	0.42	and	0.41	ed_	0.40	es_	0.39	hat	0.39
tha	0.38	ion	0.37	re_	0.37	_i_	0.36	_co	0.35	or_	0.33	t_t	0.33
you	0.33	e_a	0.33	ent	0.33	in_	0.33	_is	0.32	e_i	0.32	for	0.31
_yo	0.31	tio	0.31	_a_	0.31	thi	0.30	_be	0.30	ly_	0.30	s_a	0.29
_re	0.29	_no	0.28	nt_	0.27	t_i	0.27	_fo	0.27	_it	0.27	s_t	0.26
_ha	0.26	e_s	0.26	le_	0.26	_on	0.25	it_	0.25	her	0.25	ll_	0.25
me_	0.25	_so	0.24	n_t	0.24	_wh	0.23	ati	0.23	our	0.23	ve_	0.23
_se	0.22	s_i	0.22	ut_	0.22	ere	0.22	all	0.21	al_	0.21	ter	0.21
st_	0.21	d_t	0.21	_pr	0.21	se_	0.20	ver	0.20	not	0.20	_wi	0.20

最常用的 77 种三字母组合(去除空格)

the	1.49	ing	0.77	tha	0.52	and	0.50	hat	0.47	ion	0.45	ent	0.43
you	0.41	thi	0.38	for	0.38	ati	0.38	tio	0.38	her	0.35	ere	0.35
eth	0.34	int	0.32	our	0.28	tth	0.27	all	0.27	rea	0.26	ter	0.26
nth	0.26	ome	0.25	hin	0.25	ver	0.25	not	0.24	res	0.23	est	0.22
oul	0.22	ont	0.22	ate	0.21	uld	0.21	ers	0.21	tin	0.21	oth	0.20
pro	0.20	sth	0.20	ons	0.20	his	0.19	ith	0.19	ave	0.19	eri	0.19
sin	0.19	ess	0.18	are	0.18	hav	0.18	ist	0.18	ill	0.18	out	0.18
com	0.18	rth	0.18	ese	0.17	ore	0.17	ple	0.17	con	0.17	one	0.16
att	0.16	iti	0.16	ert	0.16	ica	0.16	ein	0.16	eto	0.16	som	0.16
han	0.15	oft	0.15	nte	0.15	ine	0.15	sto	0.15	ted	0.15	ive	0.15
ear	0.15	fth	0.15	nce	0.15	ret	0.14	ngt	0.14	ble	0.14	lin	0.14

习题

2.3.01 假设一个给定的英文字母真正是“e”的概率为0.1167，则在一个有10个单词的随机文本中没有字母“e”的概率为多少？在100个随机字符中没有“e”的概率又是多少？

2.3.02 假设一个给定的英文单词真正是“the”的概率为0.0465，则在一个有10个单词的随机文本中没有单词“the”的概率为多少？在100个单词的随机文本中没有“the”的概率又是多少？

2.3.03 假设有一种语言，它仅使用两个字符“0”和“1”。假设在一般情况下，“0”在该语言中出现的概率是1/3，而“1”出现的概率则是2/3。那么在该语言的一个长为N的随机字符流中，有$N/2$个或者少于$N/2$个“1”的概率为多少？分别考虑$N=3,6,9,12$的情况。

2.4 对仿射密码的攻击

根据字符、单词、双字母组合等频率的相关信息，我们就能给出一种对仿射密码的唯密文攻击方法，而这种方式比尝试所有可能密钥的穷举法攻击效果要好得多。

例如，假定有如下加密的文本：

JFFGJFDMGFSJHYQHTAGHQGAFDCCFP

我们的目的就是找出密钥(a,b)使得：

$$E_{a,b}(\text{明文})=\text{JFFGJFDMGFSJHYQHTAGHQGAFDCCFP}$$

由于已经去除了所有的空格，我们就不能直接利用短单词频率。然而，统计单个字母百分比：

F	20.68
G	13.79
H	10.34
J	10.34
Q	6.89
A	6.89
C	6.89
D	6.89

P	3.44
S	3.44
T	3.44
Y	3.44
M	3.44

这就使得我们认为，字母“F”就是字母“e”加密后的结果，因为，“e”在英文中是最为常用的字母而“F”密文中是最为普遍的字母。而且在英文第二个常用的字母是“t”，而这段在密文中频率处于第二位的字母是“G”，于是就可以猜测“G”是“t”加密后的结果。为了评估猜测结果的质量，我们必须首先确定密钥(a,b)，因此就有：

$$E_{a,b}(e)=F \quad E_{a,b}(t)=G$$

由前面介绍过的字母编码规则，“e”的编码为 4，“F”为 5，“t”为 19，“G”为 6。因此就有：

$$(a\cdot 4+b)\%26=5$$
$$(a\cdot 19+b)\%26=6$$

那么由以前对仿射密码选择明文攻击的讨论，就有（两式相减）

$$a\cdot(19-4)\%26=6-5=1$$

也即：

$$15\cdot a\%26=1$$

我们发现（通过穷举法或其他方式）15 模 26 的乘法逆元是 7，于是$a=7$就是我们通过猜测计算出的结果，再将它代入等式

$$(a\cdot 4+b)\%26=5$$

代入$a=7$，就可得出：

$$(7\cdot 4+b)\%26=5$$

这也就是$2+b=5$，于是（猜测）$b=3$。

也就是说，基于频率猜测出密钥是(7,3)。假如这样的话，一般的加密公式的逆则为

$$E_{a,b}^{-1}=E_{a^{-1},-a^{-1}b}$$

a^{-1}表示a关于模 26 的乘法逆元。我们已有$15\cdot 7\%26=1$的结果，因此 15 就是 7 的模 26 的乘法逆元。于是有如下计算：

$$-7^{-1}\cdot 3\%26=-15\cdot 3\%26=-45\%26=7$$

因此，$E_{7,3}$的逆就是$E_{15,7}$。在密文中应用$E_{15,7}$就可以恢复明文：

$$E_{15,7}\text{（JFFGJFDMGFSJHYQHTAGHQGAFDCCFP）}$$
$$=\text{meetmeaftermidnightinthealley}$$

这很容易就分解成 meet me after midnight in the alley。庆幸的是我们第一次猜测是正确的。

注意，在这个唯密文攻击的试验中，我们运用频率分析得出的明文比较容易接近于正确的明文。如果缺少一点运气的话，我们不得不重新猜测并决定密钥，然后解密并判断结果是否合理。

习题

2.4.01 解密由仿射密码加密的密文“VCLLCP BKLC LJKX XCHCP”。

2.4.02 解密由仿射密码加密的密文“LBBKL BJMKB QLTQW TXIKT WKIBJ AABN”。

2.4.03 解密如下密文“DBUHU SPANO SMPUS STMIU SBAKN OSMPU SS”。

第3章 置 换

一些古典密码在重要应用上存在不足，但其使用的思想在现代密码学中还有应用，比如**代替**（substitution）和**换位密码**（transposition cipher）。代替密码也就是众所周知的**暗号**（cryptograms），换位密码也叫**变位字**（anagrams）。

这两种密码的基本机制都是反复移动，代替密码在字母表范围内移动字母，换位密码在消息内变换字母的位置。从数学的角度讲，对一个集合的元素进行移动就是对该集合的一个**置换**（permutation）。置换可以进行分析，并有易理解的内部结构，我们看一些洗牌和其他确定存在的混合的例子，多少会感觉到些惊异。

顺读和倒读一样的短语，即**回文**（palindrome）结构是一个不直接与密码相关的挑战性娱乐。比较简单的有“A man, a plan, a canal, Panama ”，还有拿破仑的“Able was I ere I saw Elba”。再早一些的是彼得·希尔顿 1943 年在 Bletchley 公园[⊖] 说的“Doc, note, I dissent, a fast never prevents a fatness. I diet on cod”。

3.1 暗号：代替

无论**暗号**还是**单表代替密码**（monoalphabetic substitution ciphers）都是经常出现在日报中作为迷题的密码，从每天有数以百万计的人破解了这些难题的事实可以判断出，代替密码是不安全的。这种观点是正确的，正如一事实所阐述，大的密钥空间本身不提供安全保障。移位密码（shift cipher）和仿射密码（affine cipher）正是单表代替密码的特例。因此当我们看到代替密码被破译时，在某种意义上也就是特例被破译了。

此外，**单表代替密码**是以字母表中每个字母每次在明文（“单一字母密码”）中出现时都以同样方法加密而命名，并且明文字母的加密形式与明文字母的位置相同（“代替密码”）。

如果你不很熟悉代替密码，以下将做介绍。密钥是字母表的任意组合，当然我们需要描述和记忆这种组合的方法。一般我们可以给出一个明密**对应表**：

a b c d e f g h … x y z

C E A T X H B R … D S U

第一行行给出明文字母，每个明文字母下面对应的是它加密的密文字母。例如，a 加密为 C，b 加密为 E，c 加密为 A 的情况等等。为记住密钥，协商一段英文隐秘句子，例如

ASTITCHINTIMESAVESNINE

然后删除重复的字母（也包括空格和标点符号）

ASTICHNMEV

在后面再补充上句子中没有用到的字母：

ASTICHNMEVBDFGJKLOPQRUWXYZ

⊖ Bletchley 公园是二战时期英国密码研究机构所在地。——译者注

这就做成了一个加密表：

a b c d e f g h i j k l m n o p q r s t u v w x y z
A S T I C H N M E V B D F G J K L O P Q R U W X Y Z

当然，由于在快速解密时查找下面一行不是很方便，所以做一个单独的解密表是比较明智的选择。

举个例子，我们用这个密钥来加密如下明文：

"The history of the National Security Agency is shrouded in secrecy. Even its budget is classified."

对应密文为：

"QMC MEPQJOY JH QMC GAQEJGAD PCTROEQY ANCGTY EP PMOJRICIEG PCTOCTY CUCG EQP SRINCQ EP TDAPPEHECI."

对单表代替密码最有效的攻击是利用自然语言的使用频率：单字母，双字母组/三字母组，短语，词头/词尾等，这里仅考虑英文的情况，我们利用单字母使用频率的经验和公共英语单词的知识。最常用的单字母英文是 e 和 t，其他的字母使用频率相对来说就小得多。

这样，攻击一个密码，首先统计密文中最常出现的字母，例如在密文

QCIV XY KEO JLYYW JBRO XN KEO JKGOOK.TOK SO KX KEO AELGAE XY KBSO.KEO NBJE CGO MLSDBYT CYR KEO AXKKXY BJ EBTE.XLG JKCKO NCBG BJ KEO HOJK JKCKO NCBG.

中，按照它们的出现频率的排列次序为，

K－15，O－13，E－9，B－7，J－7，C－6，X－6，Y－6，G－5，L－3，N－3，A－2，S－2，T－2，R－2，而 D、H、I、M、Q、V、W 则出现的很少。

这样，我们可以设想"K"是"e"或"t"，"O"是两个中的另一个。首先假定 K=e 并且 O=t，得到（一部分）

QCIV XY eEt JLYYW JBRt XN eEt JeGtte. Tte St eX eEt AELGAE XY…

"Tte"在第二个句子里立刻出现了问题，似乎任何一个字母替代"T"所形成的词来引导一个句子都是不可能的。因此，改试 K=t 并且 O=e:

QCIV XY tEe JLYYW JBRe XN tEe JtGeet.Tet Se tx tEe AELGAE XY tBSe.tEe NBJE CGe MLSDBYT CYR tEe AXttXY BJ EBTE.XLG JtCte NCBG BJ tEe HeJt JtCte NCBG

由"tEe"猜想 E=h，"tX"猜想 X=o，而"XY"猜想 Y=n。这就得到

QCIV on the JLnnW JBRe oN the JtGeet.Tet Se to the AhLGAh on tBSe. the NBJh CGe MLSDBnT CnR the Aotton BJ hBTh. oLG JtCte NCBG BJ the HeJt JtCte NCBG

"Tet Se to the"猜想"get me to the"，有 T=g 并且 S=m。"JtGeet"可能是"street"，因此 J=s，G=r:

QCIV on the sLnnW sBRe oN the street. get me to the AhLrAh on tBme. the NBsh Cre MLmDBng CnR the Aotton Bs hBgh. oLr stCte NCBr Bs the Hest stCte NCBr.

由"MLmDBng"的结尾和"Bs hBgh"，猜想 B=i。由"oLr"想到 L=u。重新整理：

QCIV on the sunnW siRe oN the street. get me to the AhurAh on time. the Nish Cre MumDing CnR the Aotton is high. our stCte NCir is the Hest stCte NCir.

那么"sunnW siRe oN"猜想 W=y，R=d，N=f:

QCIV on the sunny side of the street. get me to the AhurAh on time. the fish Cre MumDing Cnd the Aotton is high. our stCte fCir is the Hest stCte fCir.

由“Cre MumDing”想到C=a，“the Hest”想到H是b。并且由单词“AhurAh”想到A=c，“Aotton”想到A=c。到这里，剩下的明文字母不多了，因为“b”已经存在，所以“MumDing”不能是M=b。经过反复几次试验后，我们可以认定M=j，D=p，现在得到：

QaIV on the sunny side of the street. get me to the church on time. the fish are jumping and the cotton is high. our state fair is the best state fair.

通过观察单字母频率得到的一点提示，再反复试验，密文就能够被破译。

如果保留了单词的边界，那么破译这样的密码就特别容易，这也是因为单词边界的保留就可以使解密者充分利用短单词的信息。流行报纸上的密语几乎都保留单词边界，否则破译这样一个密文是相当困难的。尽管如此，对于足够大量的消息，使用英文字母的统计特征，破译单表代替密码是可行的。

顺便说一下，密钥空间是巨大的，加密a有26种选择，剩下25种选择加密b，24种选择加密c，依此类推，这样就有

$$26\cdot 25\cdot 24\cdot \cdots \cdot 5\cdot 4\cdot 3\cdot 2\cdot 1 = 403\ 291\ 461\ 126\ 605\ 635\ 584\ 000\ 000$$

种不同的密钥。如果解读一个暗号真需要判断这么多种可能性的话，仅仅依靠笔纸没有人能够做出来。对密钥空间的清晰考虑很容易被错误地理解为：太小的密钥空间不好，但是大的密钥空间自身也不能保证安全。

习题

3.1.01 破译密文“PBIYGR GRQQRO KORNSRIEBRP CGJIR JKQRI IRCOGX PSKKBER QJ VREOXMQ PSLPQCIQBCG RIYGBPA PRIQRIERP, CGQAJSYA FRRMBIY UJOV LORCFP BI HCFRP CI RIJOHJSP VBKKRORIER.”其间隔是英文单词间真正的间隔。

3.1.02 破译密文“UHNDJST NO P MTFPQNVTFY HTW NSTP RJM THDJSNHI FPMITM PFKAPETQO PHS NSTJIMPKAND OYOQTGO EY UONHI QWJ EYQTO KTM DAPMPDQTM.”其间隔是英文单词间真正的间隔。

3.1.03 破译密文“EPKH RTD RTDKPDH BUQR MPKVDJ RTD PCRIK KE MPIHIRIVD PKKRQ RK CGG DGDHDJRQ IQ HKQR KERDJ CLKVD KJD OUCPRDP.”其间隔是英文单词间真正的间隔。

3.1.04 破译密文“KOS UVXW YFD YMY CFK VUUFQ KOS HNMRA EIFQC LFT KF ZNBG FPSI MK，QOMRO VUUFQJ BNRO QIMKMCD QMKOFNK NJMCD KOS BFJK RFBBFC ROVIVRKSI NCKMU ZNJK CFQ.”其间隔是英文单词间真正的间隔。

3.2　变位字：换位

简单换位密码或**变位字**都是单个字符不变而位置改变的密码。也就是说，密文是由明文适当地混序产生。自身混序的过程就是**换位**和**变位**，一段有意义的短语有意重新安排称作原文的**变位**。当然许多人在使用规则的基础上解决了变位字难题，这说明换位密码的确是不安全。

例如，通过将文本简单翻转的密钥加密如下明文

Education ought not be merely vocational training

为

GNINIART LANOITACOV YLEREM EB TON THGUO NOITACUDE

执行同样的逆过程即可解密。一个不知道的人可能无法理解它，但是这里的秘密是一个过于简单的事情，一旦被发现，密码也就被破译了。一段顺读和倒读都有意义的句子称作**回文**。但是如果我们给出大量杂乱无章的字母，比如

MOPOOMSEHCWEELNEA（来源于“please come home now”）

无论采用哪种变换都不能从英文中得到它的任何信息，有许多种方法可以将其重新整理为有意义的英文，与其他类型的密码相比，不管消息的长度是多少，这都是正确的。多种破解的问题（在密码分析攻击方面）对于某些短单词是非常明显的，比如“ant”和“tan”是变位字，还有“keep”和“peek”，“live”、“evil”和“vile”等等。

变位就是重新整理自然语言短语，使之成为完全不同但仍有意义的相同或另外一种自然语言的游戏。事实上，当用换位密码加密一个简单消息时，产生的消息块的长度至少同原消息长度相同，这是一种可能使唯密文攻击得出许多种不同破解结果的方法，这种可能性随着消息长度的增长而增大而不是减小。然而，如果多次使用一个特殊的换位加密方案，下面的多重变位将提供一种确切的唯密文攻击，这在某种程度上同一次一密的性质是类似的，同样会因为多次重复使用而受到攻击。

分组换位密码针对固定大小的分组字符操作，在每个块中对字母进行混合。在某种程度上，对所有的分组都是相同的。解密就是把混合过程逆向执行。例如，我们可以编制一个用每个n字节长的分组颠倒字符的换位密码。把一条消息分为n字符长的多条，末尾剩下的分组用任意的字符填充为n字符长，同其他的分组同样进行处理。举例来说，当n=2时，在消息中第$2k$和第$2k$+1个字符是互换的，这是一个特例。比如将如下明文

The rain in spain falls mainly in the plain

转换成密文

HT EARNII NPSIA NAFLL SAMNIYLI NHT ELPIAXN

当然，这看起来杂乱无章，但是知道了技巧，甚至通过观察就可以很容易的解密。

当旧的换位被发现时，为改变密钥，如何能够区分出混乱的明文呢？如何很细致的做到这点，并非显而易见。这个问题的困难性又一次表明，指定一类加密过程的问题可以通过依赖一个密钥的方式来简单地实现。但是，我们可以适当地忽略按照密钥的方式来描述换位这个问题，而只是去了解如何实现和如何攻击换位密码。

备注 如上所述，当对英文这样的自然语言加密时，如果简单代替密码的分组小于报文长度的一半，则这样的密码是很容易被破译的。此外，使用相同的密码加密多条消息同样是易受攻击的。尽管如此，换位仍是许多安全的现代密码的重要组成部分。

针对这些简单换位密码的基本唯密文攻击，就是试图最大程度地利用常用双字母组的连接关系，这意味着需要计算排列，使得不可能或者不可靠的双字母组的数量最小化，并且使常用双字母组的数量最大化。这种方法有时被称为**联系的方法**。值得注意的是换位密码不影响单字母频率，“e”仍保持为“e”等等，因此单字母频率对于已经确切知道是简单换位方法的密码提供不出任何有用的信息。实际上，如果密文的单字母频率本质上与其他英文相同，那么我们认为它是简单换位密码。这种方法对于短的变位字效果非常好，毕竟它是一个手工

方法的简单形式，选取出常用单词片断或小单词，然后以某种合理的方式安排剩余部分。

举个简单的例子，给出一个密文 KTICH，以下使用联系的方法。最常见的双字母组是“th”，还有“ck”。这样，我们试着寻找更小的块来代替“K”“T”“I”“C”“H”5 块，恰好是“TH”“CK”“I”。从而，取代 5！=120 种可能排列，缩减为寻找 3！=6 种排列可能。此外，由于只有一个元音，而且它必须在中间，所以仅有两种解密可能，“ckith”和“thick”，后者正是我们需要的。如果我们采取另一种不同的方式，（错误地）设想通常会存在“CH”，同时排除不可能的“KT”和“TK”，然后“I”和“CH”二者之一必须出现在“T”和“K”之间，但是很明显我们需要一个元音，也就是“I”。同样地，“CH”不能同“K”相邻，并且“TCH”和“CHT”在英文中根本不常见。但是到现在为止我们排除了所有使用“CH”的可能性，因此我们通过联系的方法判断（相当机械）“CH”不存在。

备注　注意到这是一种真正可行的概率方法。如果最有希望的可能性失败了，就必须返回，尝试可能性较小的方法，就像暗号（cryptograms）一样。

举个稍微大一点的例子，考虑密文“SEGHCAN”，有 7！=5040 种可能的排列。在用肉眼可以识别出单词前，我们想尽可能多机械地排除一部分。让我们首先通过排除不可能出现的双字母组来缩减查找范围：“hc”、“hs”、“gn”、“ae”、“sg”、“gc”、“cg”、“hg”都是不可能的。由于极少量的元音，所以在这里“gs”也是不可能的。进而，在这种情形下，破译结果的末尾不可能是“a”、“c”、“g”、“h”、“n”或者“e”。我们就可以把注意力集中在包含“sh”或“ch”的译码上。一个纯粹机械的过程表明剩余的可能性仅有 22 种（不是 5040 种）。为了能明确看出，我们将其一一列出：egachns、echngas、gechans、gechnas、geachns、genchas、gachens、gachnes、ganches、chegans、chengas、chagens、changes、chnegas、chngeas、chnages、agechns、achnges、ngechas、ngaches、nchegas、nchages，“changes”显然是可能的解密结果。

然而，对于大的变位字（分组同报文长度一样）将有许多似是而非的解密，这个方法就是失败的。例如，密文“RHETDA”（6！=720 种可能排列）能被排列为“thread”，并且也可以是“dearth”、“at herd”和“red hat”等等，我们如何能够分辨出哪一个是正确的呢？我们不能。进一步的上下文可能有所帮助，用这种方法加密的上下文可以产生大量的附加信息。因此，模糊性不能排除。这样，与许多其他的古典密码比较，当加密分组的大小和消息一样大时，解密换位密码存在比较严重的问题

但是，如果一个攻击者拥有同一密钥的两个或更多简单换位密码加密的密文，或者（相同意义）如果分组大小比消息的一半还要小，就会发生**多重变位字攻击**。这种攻击的原理是，尽管单个变位字允许有许多不同的有意义排列，但是两个变位字同时合理地呈现出超过一种排列是非常不可能的。

例如，假设收到两个密文“ESROL”和“VIERD”，并且知道是通过简单换位密码用相同密钥加密得到的。“ESROL”至少有两个较合理的解码“loser”和“sorel”，而“VIERD”也同样至少有两个解码“drive”和“diver”。然而，只有通过移动第 1 个字母到第 4 个，第 2 个到第 3 个，第 3 个到第 5 个，第 4 个到第 2 个，第 5 个到第 1 个字母的位置的同样排列，所给出“loser”和“drive”作为破解的明文更为合理。

备注　从联系方法的观点看，目标是缩减查找空间。如果我们不设法耍点聪明的话，5 件事情有 5！=120 种不同种排列，但是我们能决定忽略一些排列，对

“ESROL”包括“sr”、“lr”或“oe”　的破解

“VIERD”包括“vr”、“vd”或“dv” 的破解

由于这些都是非常不常见的双字母组。虽然它是一点单调乏味的检验，但是这样仅剩下24种可能的排列。在解密“ESROL”上，我们能够进一步排除“eo”的存在。同样的原因，在解密“VIERD”时，阻止了“vr”的出现，因此没有任何进展。但是在排除了“VIERD”以“v”或“i”结尾译码的情况后，缩减查找空间为15，并且排除“ESROL”以“o”结尾，可能排列的数目缩减到了10（由于不需要高水平的英文理解力，这些数能够机械的得到）剩下的10种可能列表足可以由一个人观察出来：

oserl rived
oresl revid
olser rdive
loser drive
esorl vired
eslor vidre
ersol veird
erosl verid
serol iverd
resol evird

在两组破解中仅有的有意义的一组英文是第4组，即“loser”和“drive”。

备注 注意到所有其他的古典密码，选择明文攻击都能够立即揭示密钥。也就是说，从现代的观点看来，简单换位密码远远不能满足密码的基本标准。尽管如此，理解唯明文攻击能够成功仍是重要的。

备注 在密码分析学中，搞清楚机械是什么和人做出判断需要些什么练习之间的区别是重要的。机器能够以较低的错误率执行重复率较高的任务，而人类做不到。因此，最大限度地机械预排序或预过滤（在需要人查看可能破解前）是主要目的。而语言更多诡辩模式的发展的确既有趣味也有潜在的作用，看来用简单语言模式获得成功的方法更具有生命力。

习题

3.2.01 破解“IIVLC”，它是使用简单换位密码加密。

3.2.02 破解“HETEM”，它是使用简单换位密码加密。

3.2.03 破解“ALBST”，它是使用简单换位密码加密。

3.2.04 破解“RIGYM”，它是使用简单换位密码加密。

3.2.05 破解“EPSILTP”，它是使用简单换位密码加密。

3.2.06 破解“TSAALPEN”，它是使用简单换位密码加密。

3.2.07 破解“EAGGAR”和“DAIREP”，它们使用相同的简单换位密码加密（变位字）。

3.2.08 破解“YLAINLF”和“ELYICCB”，它们使用相同的简单换位密码加密（变位字）。

3.2.09 破解“QHEIASUMS”和“LYGCAHIRO”，它们使用相同的简单换位密码加密（变位字）。

3.3 置换概念

直观地说，对一些事物**置换**就意味着反复地移动它们，更确切些，一个**集合** X 的**置换** f 被定义为 X 到自身的一个双射函数 f。

关于 X 的置换我们关心的问题也就是有多少个置换？如果 X 有 n 个（不同的）元素 $x_1,x_2,\cdots,x_n$ 并且 $f:X\to X$ 是 X 的一个置换，那么 $f(x_n)$ 的值有 n 种选择，$f(x_{n-1})$ 的值可以有 $n-1$ 种选择（因为它不能是 $f(x_n)$ 的值）等等，这样 n 个元素的集合总共就有 $n!$ 个置换。

这里有一个很重要的问题就是，给定的某个状态经过多少次置换后能够返回到其初始状态？这不仅与洗牌有关，也与随机数的产生和其他问题有关。

为了研究置换本身同集合的元素内容无关，只需要我们能够把它们分离开，看一下如下这个集合

$$\{1，\ 2，\ 3，\ \cdots，\ n-1，\ n\}$$

它是 n 个元素集合一个很好的例子。n 个元素的**置换群**的标准符号应该记为 S_n，S_n 也被称为 n 个元素的**对称群**。

尽管被称为“对称群”，这些群与我们所熟悉的几何中的“对称群”还不是直接相关的。同时，一个对称性的集合可以证明是一个对称群，这种情况确实可能产生。如果我们愿意把包含 n 元素的集合称为一个“几何物体”，那么对称群就是一个对称性的集合。这里的目的是用术语试图说明一些事情，但是参考引证有些不足，因此注意你所参考使用的术语。

为详细的描述 f 的操作，标记 $\{1，\ 2，\ \cdots，\ n\}$ 上的置换 f 的标准方法是将 f 有效地图形化，具体为如下列表形式：记为

$$f=\begin{pmatrix}1 & 2 & 3 & \cdots & n\\ f(1) & f(2) & f(3) & \cdots & f(n)\end{pmatrix}$$

这样，稍微改变一下符号很容易，置换

$$g=\begin{pmatrix}1 & 2 & 3 & \cdots & n\\ i_1 & i_2 & i_3 & \cdots & i_n\end{pmatrix}$$

就是满足 $g(l)=i_l$ 的一个置换。

我们总可以得到**平凡置换**

$$e=\begin{pmatrix}1 & 2 & 3 & \cdots & n\\ 1 & 2 & 3 & \cdots & n\end{pmatrix}$$

这是一个没有移动集合任何元素的置换。也就是说，对于所有的 i，$e(i)=i$。

当然，一个置换之后还可能应用另一个置换，如果 g 和 h 是两个置换，记为

$$g\circ h$$

表示先应用 h，然后应用 g。这是两个置换的**合成**或**乘积**，区分应用置换的顺序是比较重要的，通常，

$$g\circ h\neq h\circ g$$

我们来看下面的例子。任何情况下，这个符号都同函数合成的符号相一致，即对于 $1\leqslant i\leqslant n$，定义

$$(g\circ h)(i)=g(h(i))$$

作为置换定义的一个结果，即置换是集合到自身的（双射）函数，置换的合成满足结合

律：对于一个集合的所有置换 g,h,k，

$$(g\circ h)\circ k = g\circ(h\circ k)$$

事实上，对于集合的任意元素 i，置换合成的定义给出下述等式：

$$\begin{aligned}((g\circ h)\circ k)(x) &= (g\circ h)(k(x)) && ((g\circ h)\circ k \text{ 的定义，应用于 } x)\\ &= g(h(k(x))) && (g\circ h \text{ 的定义，应用于 } k(x))\\ &= g((h\circ k)(x)) && (h\circ k \text{ 的定义，应用于 } x)\\ &= (g\circ(h\circ k))(x) && (g\circ(h\circ k) \text{ 的定义，应用于 } x)\end{aligned}$$

（这个结果对无限集合也同样适用。）

对于任意置换 g，存在一个通过置换 g 执行反向操作得到的**逆置换** g^{-1}，即有

$$g\circ g^{-1} = g^{-1}\circ g = e$$

表示合成的小圆圈标志经常被省略，我们记

$$g\circ h = gh$$

就好像是普通的乘法。我们不能认为 $gh=hg$，这一点必须注意。

置换的图表记法对于计算两个置换乘积是非常有效的，例如计算：

$$\begin{pmatrix}1&2&3\\2&3&1\end{pmatrix}\circ\begin{pmatrix}1&2&3\\3&2&1\end{pmatrix}$$

我们看到这个合成对 1、2、3 的每一个都执行了什么操作。右边的置换先执行，1 变为 3，再通过第二个置换（左边的置换）到 1。相同地，2 变为 2（通过右边的置换），再到 3（通过左边的置换）。同样地，3 到 1（通过右边的置换），再到 2（通过左边的置换）。图表描述这种情况则是

$$\begin{pmatrix}1&2&3\\2&3&1\end{pmatrix}\circ\begin{pmatrix}1&2&3\\3&2&1\end{pmatrix}=\begin{pmatrix}1&2&3\\1&3&2\end{pmatrix}$$

假如我们以相反的顺序做乘法（合成），则得到不同的结果：

$$\begin{pmatrix}1&2&3\\3&2&1\end{pmatrix}\circ\begin{pmatrix}1&2&3\\2&3&1\end{pmatrix}=\begin{pmatrix}1&2&3\\2&1&3\end{pmatrix}$$

这就是置换乘积**不可交换性**的最简单的例子，也就是说，通常 $gh\neq hg$。

置换，特别是大集合上的置换，可能是难以想象的复杂事物。尽管如此，它们仍能被分为多个简单的部分，我们往下看。

首先，最简单的置换是不同长度的**循环**。一个 k **循环**是一个满足如下条件的置换 f，（对于一些数 $i_1,\cdots,i_k$）

$$f(i_1)=i_2, f(i_2)=i_3, f(i_3)=i_4, \cdots, f(i_{k-1})=i_k, f(i_k)=i_1$$

并且对于任意的不在 $i_1,\cdots,i_k$ 中 j 有 $f(j)=j$。注意 i_k 返回 i_1。如同循环的名称所表示的那样，f 在 $i_1,\cdots,i_k$ 内部循环。对此我们有一个简便的记法：即将这个 k 循环记为

$$(i_1 \quad i_2 \quad \cdots \quad i_{k-1} \quad i_k)$$

例如，与更加一般的记法相比较，

$$\begin{pmatrix}1&2&3\\2&1&3\end{pmatrix}=(1\quad 2)$$

$$\begin{pmatrix}1&2&3\\3&2&1\end{pmatrix}=(1\quad 3)$$

$$\begin{pmatrix}1&2&3\\2&3&1\end{pmatrix}=(1\quad 2\quad 3)$$

按顺序就是两个 2 循环和一个 3 循环。

与更加一般的记法不同，在循环的表示中有些不确定性：例如，

$$(1\quad 2\quad 3)=(2\quad 3\quad 1)=(3\quad 1\quad 2)$$

通常，按照循环的这种记法，则有 k 种不同的方法来表示 k 循环。那么按照这样的表示方法如下结论就非常明显了：

- 如果 g 是一个 k 循环，那么 $g^k=e$。

这就意味着应用 g 对集合操作 k 次，不会对集合产生任何真正的影响，即没有移动任何元素。循环之间是如何相互作用的呢？一般情况不是很好描述，但是，如果 $g=(i_1,\cdots,i_k)$ 和 $h=(j_1,\cdots,j_l)$ 分别是 k 循环和 l 循环，并且 $\{i_1,\cdots,i_k\}$ 和 $\{j_1,\cdots,j_l\}$ 是不相交的列表，那么 g 和 h 就可以很好的相互作用，它们可以交换，也就是在这种特殊的情况下

$$gh=hg$$

这样的循环被称作（合理地）**不相交的循环**。根据这一思想我们就有如下结论：

- 任何置换都可以表示为一些不相交循环的乘积，并且本质上只有这一种表示方法。

这里“本质上”的含义是指相同的循环按照不同的顺序相乘，我们认为它是同一种表示方法，毕竟这里的乘积是可交换的。这被称作置换的**不相交循环分解**。

知道了置换 g 的不相交循环分解，我们已经非常接近置换 g 的本质了。非常幸运，这种分解能够由一种系统的方法（有效的给出这个结论的直观证明）确定。例如，考虑

$$g=\begin{pmatrix}1&2&3&4&5&6&7\\4&3&2&5&7&6&1\end{pmatrix}$$

在重复应用 g 的情况下，我们能够描绘元素变化的轨迹。起先，让我们看一下 g 的反复应用使 1 发生了什么变化：首先 1 置换 4，然后到 5，再到 7，再到 1。由于已经返回 1，我们就已经完成了循环：我们看到 g 中存在的一个循环是

$$(1\quad 4\quad 5\quad 7)$$

接下来看另一个数，例如 2，在这个循环中没有出现。首先 2 到 3，然后到 2，这就已经完成了另一个循环。这样在 g 中也有一个 2 循环

$$(2\quad 3)$$

在这两个循环中唯一没有出现的一个数是 6，在 g 的作用下它没有变化。这样我们就得到了 g 的不相交循环分解：

$$\begin{pmatrix}1&2&3&4&5&6&7\\4&3&2&5&7&6&1\end{pmatrix}=(1\quad 4\quad 5\quad 7)(2\quad 3)=(2\quad 3)(1\quad 4\quad 5\quad 7)$$

在上面的例子中，6 被 g 置换回自身：$g(6)=6$。也就是说，6 是 g 的一个**不动点**。在我们的循环分解表示中，根本没有提到 6，暗示了 6 是不变的。但是我们也能够明确的以 **1 循环**指出不变元素。因此如果我们想明确在上面的例子中 6 发生了什么变化，我们可以记

$$\begin{pmatrix}1&2&3&4&5&6&7\\4&3&2&5&7&6&1\end{pmatrix}=(1\quad 4\quad 5\quad 7)(2\quad 3)=(2\quad 3)(1\quad 4\quad 5\quad 7)(6)$$

不相交循环分解能够用来确定一个置换被多次应用后却不产生任何实际的影响而需要重复的次数：出现在这个分解中不相交循环长度的最小公倍数。

置换的阶就是置换被多次应用后却不产生任何实际的影响所需要重复的次数（当然，这有可能和词“阶”的其他应用混淆。）这样

- k 循环的阶是 k。不相交循环乘积的阶就是各个循环长度的最小公倍数。

我们可以想像，由于在混合效果被抵消之前需要重复应用循环，所以较大阶的置换比较小阶的置换具有更佳的混合效果。按照这样的想法，如果一副牌洗得很好的话，那么重复洗牌一定次数后将返回它的初始次序！对于 52 张牌而言，这个限定的次数是 8，但是，洗好牌是不容易的。

举个例子，考查 S_7 的所有元素，确定它们作为不相交循环乘积的结构，计算每一种情况的数量，并且记录它们的阶。

首先，对 7 循环 $(i_1\cdots i_7)$ 进行计数：i_1 有 7 种选择，i_2 有 6 种选择，如此下去，但是有 7 种不同的方法标记 7 循环，因此共有 7!/7 种不同的 7 循环。

其次，6 循环 $(i_1\cdots i_6)$：i_1 有 7 种选择，i_2 有 6 种选择，等等，直到 i_6 有 2 种选择，有 6 种不同的方法标记每个 6 循环，因此共有 7!/6 种不同的 6 循环。

再次，5 循环 $(i_1\cdots i_5)$：i_1 有 7 种选择，i_2 有 6 种选择，等等，直到 i_5 有 3 种选择，有 5 种不同的方法标记每个 5 循环，因此共有 7!/2!5 种不同的 5 循环。

对于其他变化，我们再来对可表示为不相交的 5 循环和 2 循环乘积的置换进行计数。我们已经知道有 7!/2!5 种不同的 5 循环，但是对每个 5 循环的选择，与它不相交的的 2 循环只有一种选择，因此就有 7!/2!5 种不同的不相交 5 循环和 2 循环的乘积。我们注意到不相交 5 循环和 2 循环的乘积的阶是 $lcm(2,5)=10$。

同前例一样进行同样的推理，有 7!/3!4 种不同的 4 循环。

不相交的 4 循环和 2 循环有 7!/3!4·3!/2 种选择，这种乘积的阶是 $lcm(2,4)=4$。

不相交的 4 循环和 3 循环有 7!/3!4·3!/3 种选择，这种乘积的阶是 $lcm(3,4)=12$。

同前例一样进行同样的推理，有 7!/4!3 种不同的 3 循环。

不相交的 3 循环和 2 循环有 7!/4!3·4!/2!2 种选择，这种乘积的阶是 $lcm(2,3)=6$。

确定不相交的 3 循环、2 循环和 2 循环的数量是有一点困难，因为两个 2 循环是难以区别的。这样，有

$$\frac{7!}{4!3}\ \frac{4!}{2!2}\ \frac{2!}{0!2}\cdot\frac{1}{2!}$$

在最后除以 2!，是考虑两个 2 循环有 2! 个不同的顺序，它们仅是符号表示上的不同，而不是置换自本身的不同，这种置换的阶是 $lcm(2,2,3)=6$。

确定不相交的两个 3 循环数目的方法与前面类似：尽管我们在两个 3 循环的选择上好像是有序的，但是它们实际上不是有序的。这样的循环对的个数为

$$\frac{7!}{4!3}\ \frac{4!}{1!3}\cdot\frac{1}{2!}$$

在最后除以 2! 是考虑到两个 3 循环有 2! 个不同的顺序，它们仅是符号表示方法的不同，

而不是置换自身的不同，这样的置换的阶是 $lcm(3,3,1)=3$。

有 $7!/5!2$ 种不同的 2 循环，每一个的阶都是 2。

有 $7!/5!2\cdot 5!/3!2\cdot 1/2!$ 对不相交的 2 循环，在最后除以 2! 是考虑到两个 2 循环的可能顺序，这仅仅是表示方法上的影响，而不是置换自身的影响。

最后，有

$$\frac{7!}{5!2}\ \frac{5!}{3!2}\ \frac{3!}{1!2}\cdot\frac{1}{3!}$$

个不相交的 2 循环的三元组，在最后除以 3! 是考虑到 3 个 2 循环的可能顺序，这仅仅是表示方法上的影响，而不是置换自身的影响。这种置换的阶是 $lcm(2,2,2)=2$。

由前面的讨论，我们看到 7 个元素的任何置换的最大阶为 12，也就是不相交的 3 循环和 4 循环的乘积置换的阶。

举一个包含计数问题的特殊例子。在 S_{24} 中，我们来对不相交的三个 2 循环和三个 5 循环进行计数。类似于前面的讨论，这类置换的数量为

$$\frac{24!}{22!2}\ \frac{22!}{20!2}\ \frac{20!}{18!2}\ \frac{1}{3!}\cdot\frac{18!}{13!5}\ \frac{13!}{8!5}\ \frac{8!}{3!5}\ \frac{1}{3!}$$

除以两个 3! 是考虑到 3 个 2 循环的可能顺序和 3 个 5 循环的可能顺序。注意，由于 2 循环和 5 循环是可以辨别的，所以对于跟 5 循环相关的 2 循环的顺序就没有进一步考虑的必要了，如此类推。

习题

3.3.01 用不相交循环的乘积表示下列置换，并确定阶：

$$\begin{pmatrix}1&2&3&4&5\\2&5&4&3&1\end{pmatrix}$$

3.3.02 用不相交循环的乘积表示下列置换，并确定阶：

$$\begin{pmatrix}1&2&3&4&5&6&7\\2&5&4&7&1&3&6\end{pmatrix}$$

3.3.03 用不相交循环的乘积表示下列置换，并确定阶：

$$\begin{pmatrix}1&2&3&4&5&6&7\\2&3&4&7&1&5&6\end{pmatrix}$$

3.3.04 用不相交循环的乘积表示下列置换，并确定阶：

$$\begin{pmatrix}1&2&3&4&5&6&7\\6&5&4&2&7&1&3\end{pmatrix}$$

3.3.05 计算乘积

$$\begin{pmatrix}1&2&3&4&5&6&7\\7&5&1&6&4&3&2\end{pmatrix}\times\begin{pmatrix}1&2&3&4&5&6&7\\2&3&5&7&1&4&6\end{pmatrix}$$

3.3.06 计算乘积

$$\begin{pmatrix}1&2&3&4&5&6&7\\2&5&4&7&1&3&6\end{pmatrix}\times\begin{pmatrix}1&2&3&4&5&6&7\\2&3&4&7&1&5&6\end{pmatrix}$$

3.3.07 计算乘积

$$\begin{pmatrix}1&2&3&4&5&6&7\\6&5&1&7&4&3&2\end{pmatrix}\times\begin{pmatrix}1&2&3&4&5&6&7\\2&3&4&7&1&5&6\end{pmatrix}$$

3.3.08 5 个元素置换的对称群 S_5 中有多少个不同的 3 循环？

3.3.09 4 个元素置换的对称群 S_5 中有多少个不同的 4 循环？

3.3.10 通过将 S_4 中的元素表示为不同阶的不相交循环的乘积，对 S_4 中每种可能阶的元素进行计数。

3.3.11 通过将 S_5 中的元素表示为不同阶的不相交循环的乘积，对 S_5 中每种可能阶的元素进行计数。

3.3.12 S_5 的任意元素的最大阶是多少？S_{14} 呢？

3.4 洗牌

通过在一副牌上切换的方式来洗牌以及迅速洗多副牌，都可以看作是对一副牌的集合上进行置换，这些动作是可以分析的。某些分析结论可能是令人惊奇的，与洗牌一样的混合过程被用于交错卷积码（interleaving convolutional codes）。

对于一副牌最简单的**切牌**方法是选择一个随机点把一副牌一分为二，然后交换两部分。例如，一副 6 张的牌，0、1、2、3、4、5，牌可能被分为 0、1 和 2、3、4、5 两部分。然后，两部分以 2、3、4、5、0、1 顺序放在一起。对于 n 张牌，切牌就会产生如下效果

$$0,1,2,3,\cdots,n-2,n-1\to i+1,i+2,\cdots n-2,n-1,\ 0,1,2,3,\cdots,i$$

即按照整数模 n 约简的语言表述，这个过程可以看作一个函数

$$f_i:\mathbf{Z}/n\to\mathbf{Z}/n$$

这个洗牌过程就是

$$f_i(x)=(x+i)\%n$$

也就是说，对 n 张牌的一次切牌相当于模 n 加法操作。特别地，

$$f_j(f_i(x))=f_{i+j}(x)$$

就是说，两次切牌的效果与一次切牌的效果一样。特别地，因为即使你花一整天时间切一副牌，其效果也可能不会比做一次简单的切牌好，所以用那种方法切牌不会是一个彻底的洗牌过程。

一副 $2n$ 张的牌，好的**洗牌**方法包括把牌分为相等的两份

$$1,2,3,\cdots,n\qquad n+1,n+2,\cdots,2n-1,2n$$

然后像下面一样，把其中一半同另一半交错

$$n+1,1,n+2,2,n+3,3,\cdots,2n-1,n-1,2n,n\quad\text{（完美的洗牌结果）}$$

注意到顶端和底端的牌都不在它们的初始位置。有一种对于各种骗术非常有用的洗牌方法，它里面顶端和底端的牌保留在原来的位置：一个糟糕的交错的例子是

$$1,n+1,2,n+2,3,n+3,3,\cdots,n-1,2n-1,n,2n\quad\text{（不好的洗牌结果）}$$

对于一副牌，这种不好的洗牌同好的洗牌一样都是通过从一副牌的顶端和底端移牌。注意到只有一种洗牌，（或者可能有两种）不同于有参数的切牌。

命题 对一副有 $2n$ 张牌 $1,2,\cdots,2n-1$ 的完美快速洗牌是如下这样一个函数

$$f(x)=(2\cdot x)\%(2n+1)$$

即乘以 2 后再用模 $2n+1$ 约简。

证明　一方面，如果$1\leqslant x\leqslant n$，那么根据快速洗牌的定义，又由于交错把x放至这副牌的第$2x$位置。另一方面，如果$n<x\leqslant 2n$，记$x=n+i$。然后根据快速洗牌的定义，x被放至这副牌的第$(2i-1)$位置。由于$2n+1<2(n+i)<2(2n+1)$，我们重新表示这个式子为

$$f(n+i)=2i-1=2(n+i)-(2n+1)=2(n+i)\%2n+1$$

这证明了快速洗牌的过程正是乘2模$2n+1$，定理得证。♣

推论　使e为2模$2n+1$的阶，即e是满足$2^e=1\bmod 2n+1$的最小正整数。那么对一副有$2n$张的牌经过e次洗牌后，所有的牌都第一次返回到它们的初始位置。

证明　经过t次快速洗牌后，第x张牌被放到第$2^t x \bmod 2n+1$个位置上。方程

$$2^t x=x\bmod 2n+1$$

对于x=1,2,3,…,$2n$，包括一个特例x=1，即有

$$2^t=1\bmod 2n+1$$

满足上式的最小正整数为$t=e$，那么对于所有的x有$2^e x=x\bmod 2n+1$。♣

习题

3.4.01 证明对一副50张的牌恰好执行8次完美的快速洗牌后，所有的牌返回到它们的初始位置。

3.4.02 证明如果对一副52张的牌反复执行完美的快速洗牌，直到执行52次这样的快速洗牌后，所有的牌才返回到它们的初始位置。

3.4.03 确定对一副10张牌的完美快速洗牌的循环分解。

3.4.04 确定对一副12张牌的完美快速洗牌的循环分解。

3.4.05 确定对一副14张牌的完美快速洗牌的循环分解。

3.4.06 确定对一副16张牌的完美快速洗牌的循环分解。

3.4.07 确定对一副18张牌的完美快速洗牌的循环分解。

3.4.08 确定对一副20张牌的完美快速洗牌的循环分解。

3.4.09 不用完美的快速洗牌方法，而用自上而下的快速洗牌方法把12张牌分为相等的两部分，在所有张牌返回到它们初始位置前，必须重复进行多少次这种洗牌？

3.4.10 在前面的例子中，用自上而下的洗牌方法，那么快速洗牌的不相交循环分解是什么？

3.5　分组交错

这些交错的置换被用于古典换位密码，也被用于级联纠错码（concatenated error - correcting codes）。与快速洗牌过程相似，这些置换有一个很有吸引力的分析。给定两个正整数m和n，定义一个称为$m\times n$**典型分组交错**的置换，取$N=m\cdot n$。（实际上，这是一个自左向右，自上向下的交错，因为这是显而易见的。）

$m\times n$分组交错的有形描述非常的直截了当：按行写一组数0,1,2,…,$N-1$，自左向右，自上向下放入$m\times n$数组：

$$\begin{array}{cccccc} 0 & 1 & 2 & \cdots & n-1 \\ n & n+1 & n+2 & \cdots & 2n-1 \\ & & \cdots & & \\ mn-n & mn-n+1 & mn-n+2 & \cdots & mn-1 \end{array}$$

然后，自左向右，自上而下按列读取这些数：

$$0,n,2n,\cdots,mn-n,1,n+1,2n+1,\cdots,mn-n+1,\cdots,mn-1$$

我们发现 0 和 $mn-1$ 保留在原来的位置，这是一个不好的特征。但是一些其他好的特征和简单特性弥补了这个缺陷。

从交错的有形描述中，我们能够得到一个具有 $m\times n$ 阶分组交错效果的公式：给定 x，令 $x=qn+r$，其中 $0\leqslant r<n$。那么

$$qn+r=x\to q+rm$$

实际上，注意到在 $m\times n$ 数组中，x 所在的行是 x/n 的整数部分，而列为 $x\%n$。颠倒行列读取数组，以 m 代替 n。

例如，3×4 分组交错可通过创建如下数组获得

$$\begin{array}{cccc} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \end{array}$$

按列读取则为 0,4,8,1,5,9,2,6,10,3,7,11。我们能够得出这个置换的循环分解：

$$(13954)\,(267108)\,(0)\,(11)$$

通过比较，3×6 分组交错是一个 16 循环（忽略两个确定的不动点，即 1 循环 0 和 15）

$$1,3,9,10,13,5,15,11,16,14,8,7,4,12,2,6$$

命题 忽略两个不动点 0 和 $mn-1$，$m\times n$ 分组交错对如下集合

$$\{1,2,3,\cdots,mn-2\}$$

的作用就是乘以 m 而后模 $mn-1$ 约简，即 $x\to(mx)\%(mn-1)$。

证明 令 $x=qn+r$，$\leqslant r<n$，那么

$$m\cdot x=m(qn+r)=mn\cdot q+mr=(mn-1)q+q+mr=q+mr \bmod mn-1$$

命题得证 ♣

习题

3.5.01 找出 2×6 分组交错置换的循环分解。

3.5.02 找出 3×5（自左向右，自上而下）分组交错置换的循环分解。

3.5.03 找出 3×4（自左向右，自上而下）分组交错置换的循环分解。

3.5.04 找出 3×7（自左向右，自上而下）分组交错置换的循环分解。

3.5.05 证明 2×n（自左向右，自下而上）分组交错置换与完美的快速洗牌具有一样的效果。证明 2×n（自左向右，自上而下）分组交错置换与糟糕的快速洗牌具有一样的效果。

第4章 严格的密码

4.1 维吉尼亚密码

尽管仿射（affine）密码在抵抗唯密文攻击上比可怜的移位密码要好一些，但是它的密钥空间太小，因此如果使用频率分析的话，相对较小的一段密文就足以破解仿射密码（也就是找到密钥）。一次一密的思想也不会有多大帮助，因为密钥分配很困难。

维吉尼亚密码（the Vigenere Cipher）具有相对较大的密钥空间，加密和解密使用的思想与我们已见过的一些思想类似，只是在顺序上有些不同。它是一个对称密码：也就是说，知道加密密钥本质上就相当于知道解密密钥。它是一种多表加密算法：在密文的不同位置出现的字符通常不是以同样的方式加密的。

但是，维吉尼亚密码是一种**周期密码**，意味着如果两个同样的字符出现的间隔固定，并且为密钥长度的倍数，则它们将以同样的方法进行加密。这种周期特性的脆弱常会被人利用。

对于维吉尼亚密码，密钥是一个字符序列 $k=(k_1,\cdots,k_m)$，其中 m 为任意值。因此，在原理上存在无限多个密钥。明文 $x=(x_1,\cdots,x_N)$ 将被分为长度为 m 的段。如果消息的长度恰好不是 m 的倍数，则在末尾填充随机字符。我们在这里同样用 $x\%26$ 表示整数 x 模 26 的余数，加密函数 E_k 如下：

$$\begin{aligned}E_k(x_1,\cdots,x_N)=((x_1+k_1)\%26,(x_2+k_2)\%26,\cdots,(x_m+k_m)\%26,\\(x_{m+1}+k_1)\%26,(x_{m+2}+k_2)\%26,\cdots,(x_{2m}+k_m)\%26,\\(x_{2m+1}+k_1)\%26,(x_{2m+2}+k_2)\%26,\cdots,(x_{3m}+k_m)\%26,\\\cdots\\(x_{N-m+1}+k_1)\%26,(x_{N-m+2}+k_2)\%26,\cdots,(x_N+k_m)\%26)\end{aligned}$$

也就是说，密钥的第一个字符被加到明文的第 1 个，第 $(m+1)$ 个，第 $(2m+1)$ 个，第 $(3m+1)$ 个字符上（且模 26 约简），密钥的第二个字符被加到明文的第 2 个，第 $(m+2)$ 个，第 $(2m+2)$ 个，第 $(3m+2)$ 个字符上（且模 26 约简），依此类推。

注意，如果与需要加密的消息长度相比，m 的值很大，则我们就回到了一次一密的思想上了。

解密函数 D_k 和加密函数一样，但使用减法而不是加法：

$$\begin{aligned}D_k(x_1,\cdots,x_N)=((x_1-k_1)\%26,(x_2-k_2)\%26,\cdots,(x_m-k_m)\%26,\\(x_{m+1}-k_1)\%26,(x_{m+2}-k_2)\%26,\cdots,(x_{2m}-k_m)\%26,\\(x_{2m+1}-k_1)\%26,(x_{2m+2}-k_2)\%26,\cdots,(x_{3m}-k_m)\%26,\\\cdots\\(x_{N-m+1}-k_1)\%26,(x_{N-m+2}-k_2)\%26,\cdots,(x_N-k_m)\%26)\end{aligned}$$

这里对于任意整数 s,t，可以运用如下等式：

$$(s-t)\%26=(s+(26-t))\%26$$

（以及其他类似的等式），因此减法（且模 26 约简）确实是加法（且模 26 约简）的一个特殊情况。

如果我们对文本字符流的元素进行标记，我们可以给出字符对加密和解密的一个很好的公式：下标不是从 1 开始，而是从 0 开始。因此密钥为 $k=(k_0,k_1,\cdots,k_{m-1})$，则：

$$E_k(x_0,x_1,\cdots)=(y_0,y_1,\cdots)$$

其中，对于任意下标 i，有：

$$y_i=(x_i+k_{i\%m})\%26$$

下标也要模密钥长度。从这里我们可以看到密码的周期性。

例如，密钥为 $k=gopher=(6,14,15,7,4,17)$，明文

meet me in the alley after midnight, bring money and dont tell

加密后变为：

SSTAQ VOBIO IRRZT FEWZS GTMUT WVOXS XWCNQ FTSNH RUJCC AXVRZ

（所有非字母的字符都被去掉了，并且重新排列为五个字符一组）。

这是**多轮**加密的第一种情况，它可以完全用于各种目的。如果一个明文使用密钥长度为 m 的维吉尼亚密码加密，将得到的密文再次通过密钥长度为 n 的维吉尼亚密码加密，最后结果和通过直接使用密钥长度为 m 和 n 的最小公倍数 $lcm(m,n)$ 的维吉尼亚密码加密的结果一样。一方面，如果 m 和 n 是互素的，则 $lcm(m,n)$ 为 mn，且这是一个很大的密钥长度。另一方面，我们还可以使用一个更长的密钥。

我们来验证，两轮的维吉尼亚密码是一轮维吉尼亚密码，其密钥的长度是前面两轮的密钥长度的最小公倍数。实际上，使用下标从 0 开始的记号，两个密钥为：

$$k^{(1)}=(k_0{}^{(1)},k_1{}^{(1)},k_2{}^{(1)},\cdots,k_m{}^{(1)})$$
$$k^{(2)}=(k_0{}^{(2)},k_1{}^{(2)},k_2{}^{(2)},\cdots,k_n{}^{(2)})$$

其长度分别为 m 和 n，则根据上面的公式有：

$$E_{k^{(2)}}(E_{k^{(1)}}(x_0,x_1,x_2,x_3,\cdots))=(y_0,y_1,y_2,\cdots)$$

其中（对于所有的下标 i）：

$$y_i=(x_i+k^{(2)}_{i\%n}+k^{(1)}_{i\%m})\%26$$

设 $N=lcm(m,n)$。定义一个长度为 N 的密钥 $k=(k_0,\cdots,k_{N-1})$，其中：

$$k_i=(k^{(2)}_{i\%n}+k^{(1)}_{i\%m})\%26$$

然后我们来验证单轮加密 E_k 和两轮加密的效果是一样的：也就是说，我们必须验证对于任意非负整数下标 i，满足：

$$k_{i\%N}\equiv(k^{(2)}_{i\%n}+k^{(1)}_{i\%m})\bmod 26$$

这里的问题是什么？就是要验证对于所有 i，是否满足：

$$(i\%N)\%m=i\%m$$

和

$$(i\%N)\%n=i\%n$$

但是，如果满足 $m\mid N$ 且 $n\mid N$，则我们已经验证了这一点。m 和 n 的最小公倍数是使得上述成立的 N 的最小值。

选择明文攻击几乎是明显的：使用由 N 个 a（也就是 0）组成的字符串作为明文，则密

文即由密钥本身重复多次组成，总数达到个 N 字符为止。因为不知道密钥的长度，则必须经过几次尝试，才能使得 a 组成的字符串足够长以使得密文开始重复自己。但是和这些麻烦事不同，维吉尼亚密码很容易受到选择明文攻击。例如，如果密钥为“gopher”，一个选择明文为 aaaaaaaaaaaaaaaaaaaa（由 20 个“a”组成），则密文变成：

GOPHERGOPHERGOPHERGO

已知明文攻击并不比选择明文攻击复杂。如果使用密钥 k 对一个字符串 $x=(x_1,x_2,\cdots)$ 进行加密得到 $y=(y_1,y_2,\cdots)$，如果消息的长度比密钥的长度大得多，则至少我们可以通过下面的式子得到密钥 $k=(k_1,\cdots,k_m)$：

$$k_1=(y_1-x_1)\%26$$
$$k_2=(y_2-x_2)\%26$$
$$\cdots$$
$$k_m=(y_m-x_m)\%26$$

也就是说，如果我们使用一个新的密钥：

$$k'=(-x_1\%26,\cdots,-x_N\%26)$$

那么，使用密钥 k' 对密文 $y=(y_1,y_2,\cdots)$ 进行加密就会产生一定数量的密钥。与选择明文攻击中一样，从原理上说关于密钥长度还存在某些不确定性。

唯密文攻击还需要做大量进一步的准备工作。

习题

4.1.01 使用维吉尼亚密码对“meet me in the alley after midnight”进行加密，密钥为“gandolf”。

4.1.02 使用维吉尼亚密码对“meet me in the alley after midnight”进行加密，密钥为“pandora”。

4.1.03 使用维吉尼亚密码对“meet me in the alley after midnight”进行加密，密钥为“platitude”。

4.1.04 使用维吉尼亚密码对“meet me in the alley after midnight”进行加密，密钥为“horrific”。

4.1.05 证明如果 m 和 n 互素且都大于 2，则长度为 mn 的密钥的个数比长度为 m 的密钥的个数与长度为 n 的密钥的个数之积还要大。

4.1.06 为什么用具有同样长度的密钥进行多轮维吉尼亚加密没有任何意义？

4.1.07 为什么使用长度为 mn 的随机密钥的维吉尼亚加密要比分别使用长度分别为 m 和 n 的密钥的两轮维吉尼亚加密好（假设 m 和 n 互素）？

4.1.08 如果每轮的密钥长度都小于等于 12，则三轮维吉尼亚密码的最长有效密钥长度为多少？

4.1.09 使用密钥长度为 2 的维吉尼亚密码进行加密得到的密文为：

OCXJIW,JCBR JC XDQCO KFBKGDFQ GK REC XJICV,

ZBDLPB WLS DM EMJC CPLK TMOI,XLA BLLQ RBJI YKWLLB

使用穷举法对其进行解密，把它当做两个分别移动一个字符的移位密码。

4.2 最小公倍数 LCM 和最大公约数 GCD

我们不是总能用一个整数来除以另一个整数而得到的商还是整数。例如用 7 除 3 得到的商为 $2\frac{1}{3}$，而用 12 除 3 得到的商为 4。这样得到一个简单的定义：对于两个整数 d,n，如果 $d|n$ 为整数，则称整数 d **整除** n（或者说 d 是 n 的一个**因子**）。这相当于说，存在另一个整数 k，满足 $n=kd$。作为一种等价的说法，我们也可以说（等价），如果 d 整除 n，则称 n 是 d 的一个**倍数**。“d 整除 n”用符号表示为：

$$d|n$$

设 m,n 为整数，而且都是非零的整数，则 m,n 的**最大公因子** $gcd(m,n)$ 为最大的正整数 d 使得 d 既能整除 m，也能整除 n。m,n 的**最小公倍数** $lcm(m,n)$ 为最小的正整数 N 使得 N 既是 m 的倍数，也是 n 的倍数。

单词“最大”和“最小”即指这些事物具备的一种特性，但是一些结果还不清楚。我们希望证明如下一个重要特性，但如果没有进一步的工作（见后文）还无法证明它：

定理 m,n 的最大公约数 $gcd(m,n)$ 具有这样的特性：对于 m,n 的每一个因子 e 满足 $e|gcd(m,n)$。m,n 的最小公倍数 $lcm(m,n)$ 具有这样的特性：对于 m,n 的每一个倍数 N 满足 $lcm(m,n)|N$。

素数 p 就是那些不存在因子 d 的整数，其中 $1<d<p$。例如，素数序列的开始部分为：2,3,5,7,11,13,17,19,23,29,31,37,… 鉴于多方原因，通常不把 1 称为素数。我们在后面将要证明一个定理，即每一个正整数都可以分解为素数的乘积，而且这种分解是唯一的。对于小的整数，这个事实或多或少对我们来说比较熟悉，比如：

$$12=2^2\cdot 3$$

$$35=5\cdot 7$$

$$1001=7\cdot 11\cdot 13$$

$$47268=2^2\cdot 3^2\cdot 13\cdot 101$$

对于小的整数，我们可以凭直觉来进行因子分解。后面我们将更加系统地介绍整数分解的问题，但是将整数分解为素数的问题一直以来都是一个非常困难的数学问题。

如果我们已经知道两个整数 m,n 的素因子分解，那么我们就能够通过寻找它们的共同的素因子而很容易找到它们的最大公因子和最小公倍数。这不是一个最佳的方法，但是它和我们凭直觉进行因子分解的方法一样。例如，为了找到 12 和 15 的最大公因数，首先将它们分解为素数乘积：$12=2^2\cdot 3$ 和 $15=3\cdot 5$，则很容易看出它们的最大公因子为 3。

通常，对于每一个素数 p，整除 $gcd(m,n)$ 的 p 的方幂，是既整除 m 又整除 n 的 p 的方幂的最小值。因为这一点对于所有的素数都成立，所以我们可以得到最大公因子的素数分解。举一个较大的例子：

$$gcd(2^3\cdot 3^5\cdot 5^2\cdot 11,3^2\cdot 5^3\cdot 7^2\cdot 11^2)=3^2\cdot 5^2\cdot 11$$

因为 2^0 是出现的 2 的两个方幂中较小的一个，3^2 是出现的 3 的两个方幂中较小的一个，5^2 是出现的 5 的两个方幂中较小的一个，7^0 是出现的 7 的两个方幂中较小的一个，11^1 是出现的 11 的两个方幂中较小的一个。

类似地，最小公倍数是通过在 m,n 的因子分解中寻找每一个素数的两个方幂中较大的一

个得到的。

在后面我们将会看到，认识到存在一个更好的方法来计算最大公因子或者最小公倍数这一点很重要。

习题

4.2.01 求 12 和 15 的最大公因子和最小公倍数。
4.2.02 求 15 和 18 的最大公因子和最小公倍数。
4.2.03 求 15 和 21 的最大公因子和最小公倍数。
4.2.04 求 18 和 24 的最大公因子和最小公倍数。
4.2.05 求 19 和 23 的最大公因子和最小公倍数。
4.2.06 求 31 和 47 的最大公因子和最小公倍数。
4.2.07 求 24 和 36 的最小公倍数。
4.2.08 求 231 和 343 的最大公因子。
4.2.09 求 5609 和 5767 的最大公因子。
4.2.10 求 51051 和 55913 的最大公因子。

4.3 Kasiski 攻击

在任何针对维吉尼亚密码的唯密文攻击中，确定密钥的长度是一个非常基本的问题。(即使已知密钥的长度 m，并且将密文当作 m 个混杂移位密码的集合，故意的解密仍然比使用单个移位的密码要困难，因为对于两两不相邻的字符串，很难说出什么时候解密是正确的!)

针对维吉尼亚密码的一个有创造性的攻击方法是 **Kasiski 攻击**或者 **Kasiski 测试**，它的目标是准确判定密钥的长度（Friedrich Kasiski, 1863)。这种攻击决不保证一定成功，而且它最多只能获得一些有限的信息（仅仅只是密钥的长度)。实际上，Kasiski 攻击的这些特殊特征也是更严格的密码分析中所特有的，因此在这个方面，Kasiski 攻击是有代表性的。

如果明文中的两个三字母组出现的间隔是密钥长度的一定倍数，则它们在密文中将被加密成为同样的三字母组。因此，如果在密文中的两个相同的三字母组是因为这个原因而生成，则它们通过一定长度的间隔分开，这个间隔是密钥长度的整数倍数。因此我们在密文中寻找多次出现的三字母组，并确定它们的间隔距离就是密钥长度的整数倍数。

这个方法的正确之处在于，如果在明文中存在两个三字母组，它们的间隔距离为密钥长度的整数倍数，则这个倍数将会出现在密文的同样的三字母组的间隔距离列表中。但是这里至少存在两个问题。首先，如果密文太短，则同样的三字母组出现两次或两次以上，且间隔距离正好是密钥的整数倍数的几率就非常小。第二，如果密文太长，则这种几率就会比因为别的原因而使得出现同样的三字母组的几率要大得多（不因为它们是同样的三字母组的密文，其出现间隔是密钥长度的倍数)。

这种方法的一个具体实现如下。对于密文中多次出现的每一个三字母组，我们计算它们每次出现的所有间隔距离的**最大公因子**。如果这个最大公因子比 1 大，则我们列出该三字母组以及相应的最大公因子。然后我们猜测密钥的长度至少是其中一个最大公因子的一个因子。让我们来看在不同情况下 Kasiski 攻击是如何进行的：

例如，考察下列明文：

friendinthecomputerbusinesssentmeth
isiwonderwhatthenetworkserviceschec
kingmodementeredpasswordsforcrackab
ilityorfromthewallstreetjournalemai
lsnoopingisokintheeyesofthelawarece
ntusdistrictcourtdecisioninpennsylv

使用密钥长度为 2 的维吉尼亚密码加密且密钥为 *ab*，可以得到如下密文：

FSIFNEIOTIEDONPVTFRCUTIOETSTEOTNEUH
JSJWPNEESWIAUTIEOEUWPRLSFRWIDETCIED
KJNHMPDFMFNUESEEPBSTWPRESGOSCSADKBB
JLJTZOSFSONTIEXAMLTTSEFTKOVROAMENAJ
LTNPOQIOGJSPKJNUHFEZETOGTIEMAXASEDE
OTVSEITTSIDTDOVRUDFCJSJOOIOPFNOSZLW

在出现两次以上的三字母中，我们来求间隔距离的最大公因子：

TIE : 4
EOT : 146
OVR : 58
TTS : 58
UTI : 27
IED : 58
WPR : 36
JSJ : 160
KJN : 82

很明显，2（密钥长度）能够整除这些最大公因子中的大多数，但是其中一个 *gcd* 为 27。使用密钥长度为 3 的维吉尼亚密码，密钥为 *abc*，得到密文为：

FSKEOFIOVHFEONRUUGRCWSJPETUSFPTNGTI
KSJYOOFESYHBVTIGNFVWPTKTGRWKCFUCIGC
LKNHOOEGMFPTFTEERATUWPTDTHOSERBEKBD
IMKTZQRGTONVHFYAMNSUTEFVJPWROCLFOAJ
NSOQOQKNHKSPMIOVHFGYFUOGVHFNAXCRFEE
OVUTFITVRJETDQUSVDFEITKOOKNQGNOUYMX

在出现两次以上的三字母组中，我们找出间隔距离的最大公因子如下：

IOV : 147
KNH : 75
VHF : 3
FPT : 51
OVH : 147
WPT : 36

很明显，3（密钥长度）整除这些最大公因子。使用密钥长度为 4 的维吉尼亚密码，密

钥为 *abcd*，得到密文为：

FSKHNEKQTIGFONRXTFTEUTKQETUVEOVPEUJ
LSJYRNEGUWICWTIGQEUYRRLUHRWKFETEKED
MLNHORDFOHNUGUEERDSTYRREUIOSEUADMDB
JNLTZQUFSQPTIGZAMNVTSGHTKQXROCOENCL
LTPROQKQGJURKJPWHFGBETQITIGOAXCUEDG
QTVUGITVUIDVFOVTWDFELSJQQIORHNOUBLW

在出现两次以上的三字母级中，我们找出间隔距离的最大公因子如下：

TIG ：4
LSJ ：160
TKQ：107
YRR ：36
TVU ：5

这里，密钥长度（即 4）仅仅只整除所有五个 *gcd* 中的三个，还是不很明确。但是使用另一个长度仍然为 4 的不同的密钥 *abcx* 进行维吉尼亚加密，可以得到密文为：

FSKBNEKKTIGZONRRTFTYUTKKETUPEOVJEUJ
FSJYLNEGOWICQTIGKEUYLRLUBRWKZETEEED
MFNHOLDFOBNUGOEERXSTYLREUCOSEOADMXB
JNFTZQOFSQJTIGTAMNPTSGBTKQRROCIENCF
LTPLOQKKGJULKJPQHFGVETQCTIGIAXCOEDG
KTVUAITVOIDVZOVTQDFEFSJQKIORBNOUVLW

在出现两次以上的三字母组中，我们找出间隔距离的最大公因子如下：

TIG ：4
YLR ：36
FSJ ：160

这里，密钥长度（即 4）能够整除所有的 *gcd* 。因此，似乎在上一段落中的“不好的” *gcd* 是“运气不好”造成的结果。使用密码长度为 5 的维吉尼亚密码，密钥为 *abcde*，可以得到密文为：

FSKHRDJPWLEDQPTUUGUFUTKQISTUHRTNGWL
ITKZSNEGUAHBVWLEOGWAOSMVIRWKFISDJHG
KJPJQOEGPINUGUIDQCVWWPTGWFPTFVADMDF
IMKWCOSHUSMUJHAAMNVXRFGWNOVTQELFODM
LTPRSPJPJMSPMLRTIGHCETQIXHFNDAASGFI
NUWVHITVUMCUERYRUFHGITKRRIORHRNTAOZ

在出现两次以上的三字母组中，我们找出间隔距离的最大公因子如下：

UGU ：65
WLE ：40
JPJ ：75
ITK ：160

INU : 95

这里，密钥长度（即 5）能够整除所有的 *gcd* 。使用密码长度为 6 的维吉尼亚密码，密钥为 *abcdef* ，得到 *gcd* 为：

VMS : 107
VXW : 53
WPT : 36
VKI : 12

这里，密钥长度（即 6）只能整除其中两个 *gcd* 。但是使用一个长度仍然为 6 的不同的密钥 *abcdex* ，得到 *gcd* 为：

WPT : 36
VKI : 12

则这两个 *gcd* 都能够被密钥长度 6 所整除。使用密码长度为 7 的维吉尼亚密码，密钥为 *abcdefg* ，得到 *gcd* 为：

NUJ : 147
UJH : 147
ONU : 147
MKW : 11
PHW : 139

前面三个 *gcd* 可以被密钥长度 7 所整除，但是后面两个不行。使用长度为 7 的另一个密钥 *abcdefx* ，得到 *gcd* 为：

NUJ : 147
FNU : 147
UJH : 147
PHW : 139

（第三个 *gcd* ，即 139 不能被 7 整除），使用第三个密钥 *abcdxxx* ，可以得到 *gcd* 为：

OSF : 31
NUJ : 147
FNU : 147
UJH : 147
PVV : 191

31 和 191 还是不能被 7 所整除。使用长度为 8 的密钥 *abcdefgh* ，可以得到 *gcd* 为：

TIG : 40
XMK : 48
LWN : 160
ORH : 127

密钥长度 8 可以整除四个 *gcd* 中的三个。使用密钥长度为 9 的密钥 *abcdefghi* ，可以得到 *gcd* 为：

VMS : 107
BHF : 108

YAZ : 53
JWU : 75
WPT : 36
VKI : 9

密钥长度 9 可以整除六个 *gcd* 中的三个。使用密钥长度为 10 的密钥 *abcdefghij*，可以得到 *gcd* 为：

QIX : 139
BQE : 40
NYP : 160

密钥长度 10 可以整除六个 *gcd* 中的三个。使用密钥长度为 11 的密钥 *abcdefghijk*，可以得到 *gcd* 为：

GOS : 50
CIO : 87

这表明，在明文中不存在三字母组使得其间隔距离是 11 的整数倍数。因此，使用密钥长度为 11 的密钥对这种特殊的明文进行的任何加密都不会遭受 Kasiski 攻击。不幸的是，使用密钥长度为 13 的密钥 *abcdefghijklm*，我们可以得到 *gcd* 为：

FHF : 39
WMW : 35
BQO : 156
WIW : 65

密钥长度 13 可以整除所有的 *gcd*。

习题

4.3.01 假设（不真实的，但是为了简单起见）所有 26 个字符在一个长度为 N 的字符串中出现的概率是相等的。对于正整数 m，两个相同的字符串出现的间隔距离不是 m 的任何倍数的概率（作为 m 和 N 的函数）为多少？

4.3.02 假设（不真实的，但是为了简单起见）所有 26^3 个三字母在一个长度为 N 的字符串中出现的概率是相等的。对于正整数 m，两个相同的三字母组出现的间隔距离不是 m 的倍数的概率（作为 m 和 N 的函数）为多少？

4.3.03 同时参考上面两个练习题：如果我们承认，实际上一些字母和三字母组比其他的更容易出现，则两个相同的字符串出现的间隔距离不是 m 的倍数的概率是增加还是降低？

4.3.04 不使用三字母，而使用单个字符来进行 Kasiski 攻击是否合理？

4.3.05 为什么对由一串“随机”字符组成的明文的加密进行 Kasiski 攻击是一种愚蠢的行为？

4.3.06(*) 使用几个短密钥（但是它们的 *lcm* 很大）的多轮维吉尼亚密码与具有长度等于各单个密钥长度的 *lcm* 的单个“随机”密钥的单轮维吉尼亚密码相比，是否更易受到 Kasisiki 攻击？

4.3.07(*) 有一种 Kasisiki 攻击的变种：在密文中寻找那些在普通的英文中最不常见的字符，并计算这些字符出现的间隔距离的最大公因子。然后猜测密钥长度应该整除这些最大公

因子的大多数。这是合理的吗？

4.3.08(*) 如果我们对一个明文进行 Kasisiki 攻击结果意味着什么？Kasisiki 攻击如何说明一个文本是已经被加过密的？

4.3.09(*) 从根本上说为什么 Kasisiki 攻击能够工作？

4.3.10(*) 请量化密文的大小对 Kasisiki 攻击的效果造成的影响。

4.4 期望值

设 $\Omega=\{\omega_1,\cdots,\omega_n\}$ 为样本空间，其“原子”事件为 ω_i，且其概率为 p_i。在集合 Ω 上定义的一个实值函数 X 被称为一个**随机变量**。（实际上，通常允许随机变量具有更加一般的数值，例如复数或者向量，但是对我们来说，实值随机变量就足够了。）

至少根据惯例，因为通常用 f 来表示函数，所以我们用 X 来表示随机变量，这样可能与通常使用的针对（非随机？）“变量”的 x 更加协调一致。此外，传统上将 X 的值表示为 x_i（与微积分的惯例相冲突）。而且，在使用字母 E 来表示“加密”上也有些冲突。上下文会明确符号的真正含义。

对于 X 的一个可能值 x，我们将符号表示扩展为：

$$P(X=x)=P(\{\omega\in\Omega:X(\omega)=x\})$$

也就是，$X=x$ 的概率被定义为 Ω 中满足 $X=x$ 的子集的概率。此类随机变量的**期望值**被定义为：

$$E(X)=p_1\cdot X(\omega_1)+p_2\cdot X(\omega_2)+\cdots+p_n\cdot X(\omega_n)$$

当然，通过进行大量的输出结果为 $\omega_{i_1},\omega_{i_2},\cdots,\omega_{i_N}$ 的独立试验之后，它们的平均值

$$\frac{1}{N}(X(\omega_{i_1})+X(\omega_{i_2})+\cdots+X(\omega_{i_N}))$$

将非常“接近于” $E(X)$。但是这个观点和类似的思想具有同样的局限性，即概率是一种有限的频率。

从直觉上的内容看，这种思想最简单的模型来源于赌博。例如，假设 Alice 和 Bob（“A”和“B”）具有一个公平的硬币（意味着硬币的正面和反面的概率都是 0.5），且赌博规则是：如果硬币的“正面”朝上，Alice 付给 Bob 一美元，如果硬币“反面”朝上，则 Bob 付给 Alice 一美元。直觉告诉我们这是公平的，而且下面期望值的计算也确证了这一点。样本空间为 $\Omega=\{\omega_0,\omega_1\}$（下标 0 代表正面，下标 1 代表反面），每一点都具有 0.5 的概率。设 X 为指出 Alice 获胜（或者失败）的随机变量：

$$X(\omega_0)=-1$$
$$X(\omega_1)=+1$$

则 X 的期望值，即 Alice 期望的收益为：

$$E(X)=0.5\cdot(-1)+0.5\cdot(+1)=0$$

通常，在一个**公平的赌博**中，每一个人的期望值都为 0。（这种观点的关键之处是什么？可能对于概率的理解不一样。）

通常，一个期望值比极其天真的“平均”的思想要复杂得多，意识到这一点很重要。例如，假设我们在 0－10 之间随机选择一个整数，并对其进行平方。设定概率相等，则该平方

值的期望值为：

$$\frac{1}{10}0^2+\frac{1}{10}1^2+\cdots+\frac{1}{10}10^2=\frac{1}{10}385=38.5$$

通过首先计算0－10的平均值（即5.5），然后对其平方（即30.25）得到期望值的方法是错误的。

命题 设 X 和 Y 是样本空间 $\Omega=\{\omega_1,\cdots,\omega_n\}$ 上的两个随机变量，每一个元素的概率为 $P(\omega_i)=p_i$。随机变量之和 $X+Y$ 被定义如下：

$$(X+Y)(\omega_i)=X(\omega_i)+Y(\omega_i)$$

则有：

$$E(X+Y)=E(X)+E(Y)$$

证明 这可从定义直接计算得到：

$$\begin{aligned}E(X+Y)&=\sum_i p_i(X(\omega_i)+Y(\omega_i))\\&=\sum_i p_iX(\omega_i)+\sum_i p_iY(\omega_i)=E(X)+E(Y)\end{aligned}$$

于是命题得到了证明。 ♠

命题 设 X 是样本空间 $\{\omega_1,\cdots,\omega_n\}$ 中的一个随机变量，每个元素的概率为 $P(\omega_i)=p_i$。设 c 为一个常数，定义随机变量 cX 为 $cX(\omega_i)=c\cdot X(\omega_i)$，则 $E(cX)=c\cdot E(X)$。

证明 这可从定义直接计算得到：

$$E(cX)=\sum_i p_icX(\omega_i)=c\sum_i p_iX(\omega_i)=c\cdot E(X)$$

命题得证。 ♠

设 Ω 为一个样本空间，X 和 Y 是样本空间 Ω 上的两个随机变量。定义该样本空间上**乘积随机变量** XY 为：

$$(XY)(\omega)=X(\omega)Y(\omega)$$

如果对于 X,Y 的每一对 x,y 的可能值有下式成立，则这两个随机变量是**独立的随机变量**：

$$P(X=x\text{ 且 }Y=y)=P(X=x)\cdot P(Y=y)$$

（这种独立性定义是对前面有关事件独立性定义的一种解释。）

如果没有独立性的前提假设，下面的命题通常不是正确的。

命题 对于样本空间 Ω 上的两个独立的随机变量 X,Y，乘积的期望值为期望值的乘积：

$$E(XY)=E(X)\cdot E(Y)$$

证明 乘积的期望值的定义为：

$$E(XY)=\sum_{\omega\in\Omega}P(\omega)XY(\omega)$$

根据 XY 的定义，有

$$\sum_{\omega\in\Omega}P(\omega)X(\omega)Y(\omega)$$

为了证明命题的正确性，最好使用上面所引用的符号：令 x 取遍 X 的所有可能值，y 取遍 Y 的所有可能值，则我们可以根据 X 和 Y 的值，通过分组的方法重新整理期望值：

$$\sum_{x,y}\sum_{\omega}P(\omega)X(\omega)Y(\omega)$$

其中对于固定的(x,y)，内和超过了ω，且使得：

$$X(\omega)=x \text{ 且 } Y(\omega)=y$$

然后，使用新的符号来重新表示为：

$$=\sum_{x,y}P(X=x \text{ 且 } Y=y)xy$$

独立性的假设准确地说应该是：

$$P(X=x \text{ 且 } Y=y)=P(X=x)\cdot P(Y=y)$$

因此表达式变为：

$$\sum_{x,y}P(X=x)P(Y=y)xy$$

现在我们就可以将其分解为乘积，从而得到希望的结论：

$$=\sum_{x}P(X=x)x\cdot\sum_{y}P(Y=y)y=E(X)\cdot E(Y)$$

这就证明了命题。 ♣

当进行（使用同一个试验）几次独立试验时，就会引起独立随机变量的一种重要情况。设Ω为样本空间。考虑N次独立的试验，并考虑下列乘积由Ω中的有序的N元组元素组成：

$$\Omega^N=\underbrace{\Omega\times\cdots\times\Omega}_{N}$$

设X_i为Ω^N上的随机变量，其值只取决于第i次试验的结果。因此，对于$i\neq j$，两个随机变量X_i和X_j是相互独立的。

习题

4.4.01 如果缸中有 3 个红球和 7 个黑球，在 20 次试验（每次试验中无论取出什么球都放回去）中取出红球的期望值是多少？

4.4.02 如果缸中有 5 个红球和 9 个黑球，在 10 次试验（每次试验中无论取出什么球都放回去）中取出红球的期望值是多少？

4.4.03 如果缸中有 2 个红球和 13 个黑球，在 30 次试验（每次试验中无论取出什么球都放回去）中取出红球的期望值是多少？

4.4.04 投掷一枚公平硬币，正面朝上的期望次数（在任意一次背面朝上之前）为多少？

4.4.05 投掷一枚通常正面朝上的几率为 1/3 的硬币，正面朝上的期望次数（在任意一次背面朝上之前）为多少？

4.4.06 投掷一枚通常正面朝上的几率为 3/4 的硬币，正面朝上的期望次数（在任意一次背面朝上之前）为多少？

4.4.07 投掷一枚通常正面朝上的几率为 3/5 的硬币，正面朝上的期望次数（在任意一次背面朝上之前）为多少？

4.4.08 抛出一枚硬币，在出现正面朝上之前期望的抛掷次数为多少（使用公平硬币）？

4.4.09 抛出一枚硬币，在连续出现两次正面朝上之前期望的抛掷次数为多少？

4.4.10 在一个随机的字符流中，其中字母 e 出现的概率为11%，则两个 e 之间的期望距离是多少？

4.4.11 在一个随机的字符流中，其中字母 e 出现的概率为11%，则两个 ee 之间的期望距离是多少？

4.4.12 设 X 为一个随机变量，将其定义为“在 10 次抛掷一个公平硬币中正面朝上的次数”。样本空间为抛 10 次硬币的结果组成的所有 2^{10} 个不同的可能序列。X 本身的期望值为5，那么随机变量 $(X-5)^2$ 的期望值为多少？

4.4.13 抛掷硬币直到连续出现 n 次正面朝上，在这种情况下期望的抛掷次数为多少？

4.4.14(*) 在间隔 [0,1] 中“随机”选择两个实数，它们的乘积的期望值为多少？

4.5 Friedman 攻击

本节介绍几种使得一类古典密码失效的方法，因为它们很容易受到唯密文攻击。直到 19 世纪晚期，人们还广泛认为**多表（周期的）代替密码**（例如维吉尼亚密码）是安全的。但是**重合指数**[William Friedman, 1925]是一种针对周期性代替密码的确定性攻击方法。特别是它可以完全破解维吉尼亚密码：它猜测密钥长度的效率比 Kasiski 攻击要高，而且对于给定的密钥长度，它能很好地猜测出密钥，还非常适应于自动执行。实际上，在第二次世界大战中，许多机械的和电子的设备都实现了这个方法，它们在大战中起到了至关重要的作用。

周期的代替密码是容易受到攻击的一类密码。移位密码和仿射密码都是单表代替密码的例子，尽管在这两种情况下，代替具有特殊的形式。维吉尼亚密码是一种多表代替密码。

对于任意周期的代替密码，本节的方法在决定密钥长度上是有效的。我们还将探讨如何利用维吉尼亚密码的特殊脆弱性，并给出对它的一个完全的唯密文攻击。

迄今为止，在对维吉尼亚密码的唯密文攻击中，即使我们能够使用 Kasiski 攻击来很好地猜测**密钥长度**，与移位密码或者仿射密码相比较，剩下的解密任务也还是相当困难的。如果密钥长度为 m，则（理论上）存在 26^m 个可能的密钥。

当然，如果我们认为密钥不是一个由 m 个字符组成的随机字符串，而是一个例如在英语中有意义的一个短语，则有效的密钥数量将会小的多。此外，即使不知道密钥的长度，由于人们经常使用他们的狗的名字、最喜爱的足球队名称或者最敬畏的数学教授的名字作为密钥，所以有效的**密钥空间**相对较小。不需要尝试 26^m 或更多个密钥，我们只需要生成一个短得多的最有可能的密钥列表即可，这就是**字典攻击**。

但是我们希望一个唯密文攻击不仅仅依赖于**坏密钥**，即使我们较好地猜测到密钥长度为 m，我们也还必须考虑 26^m 个密钥。下列明文字符都被移动了相同的位数（对于任何给定的下标 i），这个明文片段就不容易被识别出来。

$$x_i, x_{i+m}, x_{i+2m}, x_{i+3m}, \cdots$$

也就是说，我们不能仅仅将密钥长度为 m 的维吉尼亚密码当作 m 个不同的移位密码来对待（如果能够的话，则事情变得容易的多，因为使用唯密文攻击很容易破解移位密码）。而且，我们能够根据明文语义上的连贯性，就能够说出我们是什么时候获得了真正的明文：如果一个所谓的解密给出一个明显不是英文的字符流，我们就会拒绝接受。如果我们仅仅考查明文中的一小段，则这个标准是不适用的。

例如，从下列明文中：

friendinthecomputerbusinesssentmethisiwonderwhatthenetworkserviceschec
kingmodementeredpasswordsforcrackabilityorfromthewallstreetjournalemai
lsnoopingisokintheeyesofthelawarecentusdistrictcourtdecisioninpennsylv

每隔八个字符取出一个字符，考察得到的结果是否明显是英文：

ftteentrennsobfateokeatidil

不，绝对不会。一段英文中的单字母频率应该和普通英语（至少如果此段英文足够长）中一样。（双字母和三字母肯定被打乱了。）

同 Kasiski 方法神奇地分离出一小段特殊信息一样，我们将要介绍的 Friedman 的**重合指数**每次也仅能解决问题的一小部分，这也使得我们一次可以解密一小部分。Kasiski 的方法可以被看作是 Friedman 的方法的一个粗略形式。

给定两个字符流，具有同样的长度：

$$y=(y_0,y_1,\cdots,y_N)$$
$$z=(z_0,z_1,\cdots,z_N)$$

则其**重合指数**为：

$$I(y,z)=\frac{1}{N}\sum_{i=0}^{N}\delta(y_i,z_i)$$

其中，

$$\delta(y_i,z_i)=\begin{cases}1 & (y_i=z_i)\\ 0 & (y_i\neq z_i)\end{cases}$$

因此，如果这两个字符流是相同的，则重合指数就是 1。

如果一个（或者两个）字符流是完全随机的，则我们期望重合指数大约为 $\frac{1}{26}\approx 0.038$，从直觉上来说这是比较合理。毕竟，如果所有的字符都是随机的，那么两个处于第 i 个位置的字符匹配的概率就是 1/26，则期望重合指数大约为：

$$\text{index}=\frac{\frac{1}{26}+\cdots+\frac{1}{26}}{N}=\frac{N\cdot\frac{1}{26}}{N}=\frac{1}{26}$$

如果两个字符串都具有典型英语的频率分布，那么我们就可以计算出重合指数的期望值大约就是0.067（或者稍微高一些）。下面我们将看到这个数字是怎么得到的。

命题 设 E 为一个任意的代替密码，对于两个明文字符流 y 和 z 及任意密钥 k，如果它们都使用密钥 k 进行加密，则重合指数 $I(y,z)$ 不会改变，即有：

$$I(E_k(y),E_k(z))=I(y,z)$$

证明 这个证明很容易。在明文字符流中，如果 $y_i=z_i$，则有 $E_k(y_i)=E_k(z_i)$。因为除了密钥 k 和位置点 i 以外，加密并不取决于其他任何事物。 ♣

这里有一个关于具有特定优点的重合指数的概率描述。注意，如果我们有 l 个长度为 N 的不同文本，而且我们想要计算它们中每两个之间的重合指数，那么，我们就必须比较 $l(l-1)/2$ 对文本。如果我们按照上述方法来计算重合指数，那么就意味着必须检查 $N\cdot l(l-1)/2$ 对字符是否相等。如果 N 很大，就会使得计算量很大。更合适的方法是，我们可以根据两份文本中单字符的频率，该频率也可视为概率，来计算重合指数的**期望值**，具体方法如下。

假设在范围0～-25内的字符i对应的概率为p_i，给定一个正整数N。我们考虑所有长度为N的字符流$y=(y_0,\cdots,y_{N-1})$的样本空间Ω_N，字符流$y=(y_0,\cdots,y_{N-1})$出现的概率为：

$$P(y)=p_{y_0}p_{y_1}\cdots p_{y_{N-1}}$$

也就是说，我们假设这些字符流是由一些以概率p_i生成字符i的“源”所“产生”的，因此第i个字符的生成与前面字符的生成是完全独立的。设p'_i为另一个（可能不同的）概率赋值，且Ω'_N为相应的由长度为N的字符流所组成的样本空间，字符流产生的概率为：

$$P(y)=p'_{y_0}p'_{y_1}\cdots p'_{y_{N-1}}$$

命题 在$\Omega_N\times\Omega'_N$上定义一个随机变量X如下：

$$X(y,z)=I(y,z)=y,z\text{的重合指数}$$

则X的期望值EX可以通过下面的公式计算：

$$EX=\sum_{i=0}^{25}p_i\cdot p'_i$$

证明 随机变量X可以表示为：

$$X(y,z)=\sum_{j=0}^{N-1}X_j(y,z)$$

其中，随机变量X_j被定义为：

$$X_j(y,z)=\delta(y_j,z_j)$$

而这里的δ定义为：

$$\begin{cases}1 & (y_j=z_j)\\ 0 & (y_j\neq z_j)\end{cases}$$

因为在字符流的不同位置对字符的选择是独立的，所以这些随机变量都是独立的。因此，跟上面观察的一样，期望值可以计算为：

$$EX=\sum_{j=0}^{N-1}EX_j$$

而且每一个随机变量X_j确实是δ在$\Omega_N\times\Omega'_N$上的期望值。我们可以计算得到这个结论。

两个字符y_1和z_1都为0（因此相等）的概率为$p_0p'_0$，都为1（因此相等）的概率为$p_1p'_1$。一般地，它们都为i（因此相等）的概率为$p_ip'_i$。这些不同的事件，即都等于不同的字符i，都是独立的，因此$y_1=z_1$的概率为如下和式：

$$P(y_1=z_1)=\sum_{i=0}^{25}p_ip'_i$$

这就完成了证明。 ♣

上述结论也表明，需要定义两个特定的字符流$y=(y_0,\cdots,y_{N-1})$和$z=(z_0,\cdots,z_{N-1})$的平均重合指数，这个定义由如下式子给出：

$$p_i^y=\frac{\text{y中字符}i\text{的数量}}{N}$$

$$p_i^z=\frac{\text{z中字符}i\text{的数量}}{N}$$

并且：

$$I_{\text{avg}}(y,z)=\sum_{i=0}^{25}p_i^y p_i^z$$

特别地，重合指数的这种定义对于单个字符流具有不同的意义，它揭示了一组频率 p_i 的某些特性：

$$I_{\text{avg}}(y)=\sum_{i=0}^{25}(p_i^y)^2$$

这个重合指数的概率形式可以被解释为，频率相同字符流 y 和 z 的“典型”指数。（实际上，我们已证明它是取遍具有同样频率和具有那种长度的样本空间的期望值。）

命题 如果对两个字符流使用具有相同密钥的换位密码加密，分别具有概率 $\{p_i^y\}$ 和 $\{p_i^z\}$ 的两个字符流 y 和 z 的平均重合指数 $I_{\text{avg}}(y,z)$ 是不变的。

证明 在一组字符流中，通过重新排序后，不同的字符出现的次数是不会变化的，这是所有的换位密码的共同特性。频率 p_i^y 和 p_i^z 因此也不会变化。 ♣

因此，如果我们希望使用重合指数的这个期望值，我们就可以把有关一个字符流的信息压缩在恰好有 26 个概率（频率）值的表中，使得恰好每一个概率值通过每一个字符流。

如果字符流 y 是英文，或者是通过一个简单的代替密码从英文加密得到，那么我们就能大致知道每个字符出现的频率 p_i：“e”出现的概率大约是 0.11，等等。使用上述的公式，以及概率值统计表，简单计算即可得到：

$$I_{\text{avg}}(y)\approx\sum_{i=0}^{25}p_i^2\approx 0.064$$

特别地，这个结果对于英文明文的“随机片段”也是正确的。

从另一个方面来看，对于由真正随机的源所生成的字符流 y，所有字符出现的概率为 $p_i=1/26$，期望的重合指数大约为：

$$I_{\text{avg}}(y)\approx\sum_{i=0}^{25}p_i^2=26\cdot\left(\frac{1}{26}\right)^2=\frac{1}{26}\approx 0.0385$$

相反地，y 如果是一个字符流，而 $I_{\text{avg}}(y)$ 并不接近于 0.067，则我们就会怀疑 y 到底是一个英文文本还是一个随机片断。在实际中，使用维吉尼亚密码对英文加密得到的字符流 v 的平均重合指数为：

$$I_{\text{avg}}(v)\approx 0.047$$

这比上述的真正的随机值要大一些，但是比英文明文值要小。

表达式：

$$I_{\text{avg}}(y)=\sum_{i=0}^{25}(p_i^y)^2$$

和

$$I_{\text{avg}}(y,z)=\sum_{i=0}^{25}p_i^y p_i^z$$

也具有几何上的解释，这就会明确它们的具体动作。特别地，对于每一个字符流 $y=(y_0,\cdots,y_{N-1})$，我们结合频率的 26 维**向量**

$$p^y=(p_0^y,p_1^y,\cdots,p_{25}^y)$$

其中，

$$p_i^y = \frac{\text{y中字符}i\text{的数量}}{N}$$

那么，

$$I_{\text{avg}}(y) = \sum_{i=0}^{25} (p_i^y)^2$$

就是这个向量的**范数**。对另一个字符流 z（没有必要和 y 的长度一样长），结合频率向量 p^z，平均重合指数

$$I_{\text{avg}}(y,z) = \sum_{i=0}^{25} p_i^y p_i^z$$

就是两个频率向量的**内积**。

命题 一般情况总有下式成立，

$$0 \leqslant I_{\text{avg}}(y,z) \leqslant \sqrt{I_{\text{avg}}(y)}\sqrt{I_{\text{avg}}(z)}$$

如果 $I_{\text{avg}}(y)$ 和 $I_{\text{avg}}(z)$ 都接近于神奇的数字 0.064，且如果频率向量 p^y 和 p^z 之间的夹角很小，则平均指数 $I_{\text{avg}}(y,z)$ 也接近于 0.064，而且将会随着该夹角的增大而降低。

证明 命题中不等式就是频率向量的 Cauchy-Schwarz-Bunyakowsky 不等式。那样，我们得到向量 p^y 和 p^z 之间夹角的余弦公式为：

$$\cos\theta = \frac{I_{\text{avg}}(y,z)}{\sqrt{I_{\text{avg}}(y)}\sqrt{I_{\text{avg}}(z)}}$$

频率在任何情况下都是非负数，因此有 $I_{\text{avg}}(y,z) \geqslant 0$。 ♣

现在我们回到确定周期代替密码的密钥长度的攻击问题上来。假设密文为 $y=(y_0,y_1,y_2,\cdots)$ 是用一个周期代替密码加密得到的，我们希望检验其周期是否为 m。对于任意字符流 $y=(y_0,y_1,\cdots)$ 和正整数 l，设

$$y^{(+\ell)} = (y_\ell, y_{\ell+1}, y_{\ell+2}, \cdots)$$

也就是说，$y^{(+\ell)}$ 是字符流 y 前移 ℓ 位得到的。如果周期长度为 m（或者 ℓ 的一个因子），那么字符流 y 和移位后的字符流 $y^{(+\ell m)}$ 是用同样的周期代替密码进行加密的（且使用同样的密钥）。因此，如果该周期整除 ℓ，我们就可以得到：

$$I(y, y^{(+\ell)}) \approx 0.067$$

反之，如果该周期长度不整除 l，则我们得到：

$$I(y, y^{(+\ell)}) \approx 0.047(\text{或更小})$$

注意，我们不希望移动 0 位，因为这样指数就会是无意义的“1”。在那种情况下，因为英语实际上不是随机的，因此相邻的两个字符并不是完全独立的（见前面的双字母统计），即使我们怀疑一个具有短密钥例如长为 3 的维吉尼亚密码，使用 $l=3$ 也不是明智的行为，因为这样重合指数将会很低。因此，我们应该尝试至少移位 $l \geqslant 3$ 位，并计算重合指数，而不是寻找那些为周期倍数的较高值。

例如，对于一段经过过滤的 980 个字符组成的未加密老式电子邮件，针对不同的移位位数计算重合指数，为了便于理解将得到的重合指数值乘以 100：

移位	重合指数×100
3	5.93
4	6.96
5	7.48
6	6.46
7	6.16
8	8.12
9	6.9
11	7.84
12	7.64
13	5.99
14	6.1
15	5.8
16	6.95
17	6.64
18	6.86
19	5.2
20	6.14
21	6.56

这里可以看出指数中的波动，样本越大则波动越小。

反之，对于同样的明文使用密钥为 *prognosticate*（长为 13）的维吉尼亚密码，计算密文自身移位后的重合指数，我们得到：

移位	指数×100
1	4.69
2	4.8
3	3.48
4	4.09
5	2.56
6	2.97
7	4.62
8	4.32
9	4.73
10	4.12
11	4.23
12	4.13
13	5.99
14	3.2

15	4.24
16	4.46
17	3.94
18	3.43
19	3.22
20	3.22
21	4.58
22	4.07
23	3.03
24	4.28
25	3.97
26	6.6
27	3.04
28	3.88
29	3.57
30	4.52
31	3.47
32	5.06
33	3.69
34	2.85
35	4.23
36	3.81
37	3.92
38	3.71
39	6.69
40	3.82

在上表中从上向下观察数据列，在移位数为 13（指数值为 5.99）、26（指数值为 6.6）、32（指数值为 5.06）和 39（指数值为 6.69）处，指数值均大于 5.0。这个模式是很清楚的："32"是反常现象，其他均为 13 的倍数。因此我们得出结论，密钥长度是 13 的倍数（实际也是如此）。因此，筛选出指数等于或者低于 4.7 对应的位移似乎是一个比较稳妥的处理方法。

对于同样的文本，使用密钥"*xyzqwe*"进行维吉尼亚加密，得到指数如下：

移位	指数×100
6	6.46
12	7.64
18	6.86
24	4.91
30	7.26
36	8.26
42	6.82
48	7.72

密钥长度相对于文本大小（1000 个字符）来说比较短，因此这个攻击可以毫不含糊地给出密钥长度为 6。

对于同样的文本，使用密钥“ *praxisperfect* ”进行维吉尼亚加密，得到指数如下（考察移位小于 50 的情况）：

移位	指数×100
3	5.32
6	5.03
13	5.99
26	6.6
27	5.03
32	4.85
39	6.69
49	4.72

位移为 13 的倍数时的特征比较明显，因此我们认为密钥长度是 13。

根据同样的假定，对于一个长密钥“ *praxisperfectoyster* ”，得到指数如下（考察移位小于 50 的情况）：

移位	指数×100
7	5.65
19	5.2
33	4.75
38	6.47
48	5.79

19 的倍数是明显的选择。

根据同样的假定，对于一个更长的密钥：

asdfgqwertzxcvbyuiopqwertkjhyuixwqpty

（长度为 37），得到指数如下（考察移位小于 80 的情况）：

移位	指数×100
15	4.97
16	5.08
20	5.31
22	4.9
26	5.03
37	5.51
48	4.72
59	4.99
63	4.79
74	7.94
78	4.76

这里大约只有 26 个字符使用同样的移位进行了加密，而位移为 37 特别是 74 时重合指数的

特征比较明显。因此认为密钥长度为 37。

下面，我们来考查如何利用 Friedman 指数来确定维吉尼亚密码的密钥 $k=(k_0,k_1,\cdots,k_{m-1})$，假设对于密钥长度，我们已经按上述方法确定了一个可能性非常大的值 m。

我们截取密文的一些片断：

$$y^{(0)}=(y_0,y_m,y_{2m},\cdots)$$
$$y^{(1)}=(y_1,y_{1+m},y_{1+2m},\cdots)$$
$$y^{(2)}=(y_2,y_{2+m},y_{2+2m},\cdots)$$
$$y^{(3)}=(y_3,y_{3+m},y_{3+2m},\cdots)$$
$$y^{(4)}=(y_4,y_{4+m},y_{4+2m},\cdots)$$
$$\cdots$$
$$y^{(m-1)}=(y_{m-1},y_{(m-1)+m},y_{(m-1)+2m},\cdots)$$

如果密钥长度确实是 m（或者是 m 的一个因子），则对于每一个指数 j，密文片断 $y^{(j)}$ 是根据明文中对应的片段用密钥为 k_j 的移位密码进行加密所得到的。回想一下，移位密码具有下列特性，即对于任意两个整数 s,t：

$$E_s\circ E_t=E_{s+i}$$

因此，对于任意两个指数 i,j，两个字符流 $y^{(i)}$ 和 $E_{k_i-k_j}y^{(j)}$ 为通过移位密码 E_{k_i} 对英文文本（或者一个随机片断）进行加密后的结果。

特别地，差值 k_i-k_j 应该是在范围 0～25 之间的整数 t，对于这样一个值，重合指数

$$I(E_t(y^{(i)}),y^{(j)})$$

非常接近于神奇的数字 0.064，而对于 t 的其他值，这个指数应该为 0.047 或者更低。但是，因为实际上相邻的字符之间必然存在一定的关系，所以我们应该使用上述记号来计算下式：

$$I(E_t(y^{(i)}),y^{(j)(+3m)})$$

意味着将密文片段 $y^{(j)}$ 向前移位（假设的）密钥长度的 3 倍。

或者，我们可以考虑使用重合指数的平均形式 $I_{\text{avg}}(y,z)$。

例如，还是使用上述 980 个字符的文本，使用维吉尼亚密码及密钥“*abcxq*”进行加密，假设密钥长度为 5，计算：

$$I(E_t(y^{(0)}),y^{(1)(+15)})$$

对于 0 到 25 之间的 t（且忽略任何指数低于 5.5 的情况），我们发现：

移位	指数×100
1	7.77
2	6.21
12	6.21
21	6.21
23	5.69

这样可以得出，在第零个和第一个片断之间的相对位移 k_1-k_0 的最佳位移值为 1，这和实际情况完全一致。根据同样的假定，我们再来计算

$$I(E_t(y^{(0)}),y^{(2)(+15)})$$

来猜测 k_2-k_0，我们得到：

移位	指数×100
2	6.72
8	6.72
17	6.21

这个结果对于 k_2-k_0 来说相当模棱两可。但是为了从另一个方面得到 k_2-k_0 的信息，我们还可以通过下面的计算来猜测 k_2-k_1：

$$I(E_t(y^{(1)}),y^{(2)(+15)})$$

通过计算得到：

移位	指数×100
1	8.8
8	5.69
17	6.73

对于 k_2-k_1，1 是最好的猜测，另外两个值仍是似是而非的。

因此，在这个例子中，k_1-k_0 几乎可以确定是 1，而 k_2-k_0 在 2、8、17 之中，且 k_2-k_1 极有可能也是 1，尽管它也有可能是 8 或 17。同时考察这些结论，因为一定有：

$$(k_2-k_0)-(k_1-k_0)=k_2-k_1$$

所以我们可以计算得出，8 和 17 是不可能的。因此，通过排除法，我们已接近（正确的）结论了，即密钥的头三个字符 k_0,k_1,k_2 就具有形式：$i,i+1,i+2$。

继续分析例子，计算：

$$I(E_t(y^{(0)}),y^{(3)(+15)})$$

得到：

移位	指数×100
19	6.2
23	6.72

这个结果还不很明确，只能说明 k_3-k_0 为 19 或者 23。计算：

$$I(E_t(y^{(1)}),y^{(3)(+15)})$$

得到：

移位	指数×100
8	6.72
11	7.24
22	6.72

这个结果本身也还是不明确的，只能说明 k_3-k_1 为 8、11 或 22。但是我们已经得出结论（至少是暂时性的），即差别 k_1-k_0 为 1，因为：

$$(k_3-k_1)+(k_1-k_0)=k_3-k_0$$

唯一都可以满足条件的情况是 $k_3-k_0=23$，这确实是正确的结论。为了证实这一结果，我们进一步计算：

$$I(E_t(y^{(2)}),y^{(3)(+15)})$$

得到：

移位	指数×100
2	5.69
6	7.25
11	5.69
16	5.69
21	5.18
24	6.21

这不能很好地确证（因为“21”也是正确的!），但是 5.18 太大而似是而非。另外一些较大的值与早先的数值不能比较。

让我们来做最后一个差值计算。在这个例子中，为了得到 k_4，我们计算：

$$I(E_t(y^{(0)}), y^{(4)(+15)})$$

得到：

移位	指数×100
9	5.68
16	5.16
17	7.23
18	5.68
20	5.68
22	6.19

对于 $k_4 - k_0$，看起来最好的相对移位是 17 位，但是我们私下知道这个相对移位为 16，至少是达到我们极限的有吸引力的数。但是因为这个极限实际上要小得多，我们在这里得到的结论还不是很明确。计算：

$$I(E_t(y^{(1)}), y^{(4)(+15)})$$

得到：

移位	指数×100
0	5.68
5	6.2
15	7.75
21	5.17
22	5.17
24	6.2

因为 $k_1 - k_0 = 1$，为了满足兼容性，所有可能性都被排除，除了 $k_4 - k_0 = 16$ 和 $k_4 - k_1 = 15$（也是正确的）。为了求得证实，我们计算：

$$I(E_t(y^{(2)}), y^{(4)(+15)})$$

希望得到 $k_4 - k_2$ 的值为 14：

移位	指数×100
8	5.17
11	5.17
14	6.21

15	6.21
20	5.17
21	6.21

这样的结果仍然不明确，但是至少 $k_4 - k_2 = 14$ 是可能的。最后，我们试试：

$$I(E_t(y^{(3)}), y^{(4)(+15)})$$

为了整个计算的一致性，希望能得到 $k_4 - k_3$ 的值为 19：

移位	指数 ×100
1	5.18
5	5.18
9	5.18
15	5.18
19	7.25
23	6.21

备注 上面的计算过程中产生的不确定性完全是意料之中的。考虑到样本的大小为 980 个字符，在密文“片断”中大约有 196 个字符是使用简单移位密码进行加密的。事实上，对于（假设的）密钥长度为 5 的情况，我们总共得到 10 个指数表是有帮助的。

现在，我们已经完成了对维吉尼亚密码的纯密文攻击。通过使用 Friedman 指数，我们知道密钥具有如下形式：

$$k = (k_0, k_0 + 1, k_0 + 2, k_0 + 23, k_0 + 16)$$

因此，剩下的复杂度从本质上说，在于对整个明文进行的简单移位加密，因为只剩下了一个未知参数 k_0。通过简单地观察具有此形式的 26 个可能的解密过程，我们可以利用穷举法来破解 k_0。此外，这个最后的穷举法可以在一定程度上自动进行，针对一些具有有限种类的密钥和英文明文文本字符流，计算 D_k(密文) 的重合指数，并观察 k_0 的哪一个值具有最高的重合指数。在目前的例子中，计算“一般英文”的字符流的重合指数，其中 k_0 在范围 0～25 内，密钥具有上述特殊形式，可以得到：

移位	指数 ×100
0	7.64
4	4.85
7	5.06
12	5.06
25	5.57

这表明，当 k_0 为 0 时，（使用密钥的这种特殊形式）通过 D_k 解密的密文的频率通常非常类似于那些普通的英文。这是正确的。如果得到的数更加不确定，我们将不得不进行解密尝试以看哪一个消息有意义。

习题

4.5.01 考查只有两个字符“0”和“1”的字母表，在某种语言中，这两个字符出现的概率分别为 2/3 和 1/3。如果“0”和“1”出现的概率为 1/2，则称由它们组成的字符流是随机的。对于由“0”和“1”组成（长度相等）的字符流 y, z，定义其重合指数 $I(y, z)$。

4.5.02 在前面一个问题中，两个随机字符流的指数的期望值 E_{rand} 为多少？对于那种语言而言，两个字符流的指数的期望值 E_{lang} 为多少？

4.5.03 在前面一个问题中，两个长度为 5 的随机流的指数与 E_{lang} 一样大的概率为多少？如果长度为 10 呢？

4.5.04 在前面一个问题中，从该语言中得出的两个长度为 5 的字符流的指数和 E_{rand} 一样小的概率为多少？如果长度为 10 呢？

4.5.05 解释为什么平均指数 $I_{\text{avg}}(y, y^{+t})$ 在猜测周期代替密码的周期时没有用处？

4.5.06 解释在对维吉尼亚密码的唯密文攻击中，为什么平均指数 $I_{\text{avg}}(y^{(i)}, y^{(j)})$ 对于猜测相对移位 $k_i - k_j$ 是有用的？

第5章 概率问题

这一章并不作为结论，但对概率问题及现象进行比较透彻的观察。特别我们引入了*方差*作为依赖于随机变量的另外一个基本量（除了先前讨论的期望值）。这使得我们得以给出大数定律的一个简单例子，它的思想实质就是当执行的试验次数越来越多时，结果真正接近于期望值。在这一论断中数量的限制是非常重要的，我们后面将研究它们。

我们给出的关于大数定律特殊情况的证明，用到了车贝雪夫不等式，这个不等式本身就是大数定律的一个更加量化并且明显的形式。不太严格地讲，车贝雪夫不等式告诉我们越来越多样本的结果是怎样快速地逼近对应随机变量的期望值的。举个例子，这个不等式告诉我们，在抛掷 $2n$ 次硬币的试验中正面朝上的次数是如何接近 n 的。当然没有理由期望它恰好等于 n，但我们有一个直觉，它应该非常接近。

5.1 生成函数

设 X 为由所有有序的 0、1 n 元组构成的概率空间 Ω 上的随机变量，并且它在每个 n 元组上的值为该 n 元组中 1 的个数。假设 1 出现的概率为 p 而 0 出现的概率为 q（当然 $p+q=1$），并且在序列的不同位置 0 和 1 是独立的。***X* 的期望值是多少？**

这是一个比我们后面要讨论的方法还要容易的方法，而且是一个直觉上的方法，但这个问题给了我们一个相对简单的机会来表明生成函数计算方法的应用。

首先，按照 1 出现的总次数，我们可以对概率进行分组。在 n 个位置中出现 k 个 1 的概率为 $\binom{n}{k}$。根据假设的独立性，任何特定的有 k 个 1 和 $n-k$ 个 0 的概率是 $p^k q^{n-k}$。因此，刚好出现 k 个 1 的概率为：

$$P(X=k)=\binom{n}{k}p^k q^{n-k}$$

其期望值是：

$$EX=\sum_k P(X=k)\cdot k=\sum_{k=0}^{n}\binom{n}{k}p^k q^{n-k}\cdot k$$

这里有一个小技巧。在这个表达式中用变量 x 代替 p 则可得到：

$$\sum_{k=0}^{n}\binom{n}{k}x^k q^{n-k}\cdot k$$

由微分的知识我们知道：

$$\frac{\partial}{\partial x}x^k=k\cdot x^{k-1}$$

两边同乘以 x，使得指数还是 k：

$$x\frac{\partial}{\partial x}x^k=k\cdot x^k$$

因此，在上面的和式中，利用 $k\cdot x^k$ 部分的导数表示，我们就可得到：

$$\cdots=\sum_{k=0}^{n}\binom{n}{k}x\frac{\partial}{\partial x}x^k q^{n-k}$$

由于导数的和等于和的导数，因此有

$$x\frac{\partial}{\partial x}\sum_{k=0}^{n}\binom{n}{k}x^k q^{n-k}$$

注意上面的和式可认为是 $(x+q)^n$ 的二项展开式，因此转化为

$$x\frac{\partial}{\partial x}(x+q)^n=xn(x+q)^{n-1}$$

将 x 再替换为 p，则 X 的期望值可表示为：

$$EX=pn(p+q)^{n-1}=pn\cdot 1^{n-1}=pn$$

备注 这个结果非常直接的，所以没有什么可惊讶的。X 是随机变量 X_i 的和，这里 X_i 表示在序列的第 i 个位置出现 1 的数目。每个 X_i 的期望值很容易计算：

$$EX_i=p\cdot 1+q\cdot 0=p$$

和的期望值等于每个期望值的和，因此我们很容易计算得到：

$$EX=E(X_1+\cdots+X_n)=EX_1+\cdots+EX_n=\underbrace{p+\cdots+p}_{n}=pn$$

这个方法对于这个特定的例子来说是比较简单的，但是我们阐述的生成函数方法是非常重要的。

习题

5.1.01 利用公式

$$x^n+\binom{n}{1}x^{n-1}y+\binom{n}{2}x^{n-2}y^2+\cdots+\binom{n}{n-1}xy^{n-1}+y^n=(x+y)^n$$

计算和

$$1\cdot\binom{n}{1}+2\cdot\binom{n}{2}+3\cdot\binom{n}{3}+\cdots+(n-1)\cdot\binom{n}{n-1}+n$$

5.1.02 计算和

$$1^2\cdot\binom{n}{1}+2^2\cdot\binom{n}{2}+3^2\cdot\binom{n}{3}+\cdots+(n-1)^2\cdot\binom{n}{n-1}+n^2$$

5.1.03 计算和

$$1^2\cdot\binom{n}{1}+2^2\cdot\binom{n}{2}+3^2\cdot\binom{n}{3}+\cdots+(n-1)^2\cdot\binom{n}{n-1}+n^2$$

5.1.04 利用条件

$$|x|<1,\quad 1+x+x^2+x^3+\cdots=\frac{1}{1-x}$$

计算级数和

$$x+2x^2+3x^3+4x^4+\cdots$$

5.1.05 计算级数和 $x+2^2x^2+3^2x^3+4^2x^4+\cdots$。

5.1.06 计算级数和 $x+2^3x^2+3^3x^3+4^3x^4+\cdots$。

5.1.07 有一枚硬币，设正面朝上的概率为 p，设 X 是随机变量，表示正面朝上的次数为 2 时所需要投掷的次数，求 X 的期望值。

5.2 方差、标准差

设 X 为由所有有序的 0、1 n 元组构成的概率空间 Ω 上的随机变量，并且它在每个 n 元组上的值为该 n 元组中 1 的个数。假设 1 出现的概率为 p 而 0 出现的概率为 q(当然 $p+q=1$)，并且在序列的不同位置 0 和 1 是独立的。我们已经知道 X 的期望值是一个直觉上比较合理的数 pn，**那么 X 的方差是多少呢**？

设随机变量 X 的期望值（也称为期望）为 μ，则均方差 σ 定义为

$$\text{方差}(X)=\sigma^2(X)=E((X-\mu)^2)$$

（是的，这里确实是 σ 的平方）也就是说，σ^2 是随机变量 $(X-\mu)^2$ 的数学期望值，σ 本身是**标准差**。

为了计算，可以应用数学期望的性质作一点简化，μ 在这里只看作常数。

$$\begin{aligned}\sigma^2=E((X-\mu)^2)&=E(X^2-2\mu X+\mu^2)=E(X^2)-2\mu E(X)+\mu^2\\&=E(X^2)-2\mu\cdot\mu+\mu^2=E(X^2)-\mu^2\end{aligned}$$

这里对于任何随机变量 X 都是正确的，因为我们不能使用它的特例的任何属性。

所以我们计算随机变量的方差，需要先计算 $E(X^2)$：

$$E(X^2)=\sum_{k=0}^{n}P(X=k)\cdot k^2$$

通常，使得在 n 元组中恰好出现 k 个 1 的情形有 $\binom{n}{k}$ 种方法，每种方法出现的概率为 p^kq^{n-k}。所以，

$$E(X^2)=\sum_{k=0}^{n}\binom{n}{k}p^kq^{n-k}\cdot k^2$$

这非常像在上节计算数学期望值时的表达式，但是现在我们要用 k^2 代替 k，我们看一下结果，由于

$$x\frac{\partial}{\partial x}x^k=kx^k$$

重复这一运算则有

$$x\frac{\partial}{\partial x}\cdot x\frac{\partial}{\partial x}x^k=k^2x^k$$

在表达式 $E(X^2)$ 中，用 x 代替 p，计算

$$\sum_{k=0}^{n}\binom{n}{k}x^k q^{n-k}\cdot k^2=\sum_{k=0}^{n}\binom{n}{k}x\frac{\partial}{\partial x}\cdot x\frac{\partial}{\partial x}x^k q^{n-k}$$
$$=x\frac{\partial}{\partial x}\cdot x\frac{\partial}{\partial x}\sum_{k=0}^{n}\binom{n}{k}x^k q^{n-k}=x\frac{\partial}{\partial x}\cdot x\frac{\partial}{\partial x}(x+q)^n$$

这里我们利用了二项展开式。进一步求导得

$$x\frac{\partial}{\partial x}\cdot x\frac{\partial}{\partial x}(x+q)^n=x\frac{\partial}{\partial x}(x\cdot n(x+q)^{n-1})=x(1\cdot n(x+q)^{n-1}+x\cdot n(n-1)(x+q)^{n-2})$$

用 p 代回 x，因 $(p+q)=1$，得到：

$$E(X^2)=p(1\cdot n(p+q)^{n-1}+p\cdot n(n-1)(p+q)^{n-2})=p(n+p\cdot n(n-1))$$

那么

$$\sigma^2=E(X^2)-\mu^2=p(n+p\cdot n(n-1))-(pn)^2=pn+p^2n^2-p^2n-p^2n^2=p(1-p)n$$

也就是说，**标准差** σ 的表达式为：$\sigma=\sqrt{p(1-p)n}=\sqrt{\text{方差}}$

习题

5.2.01 设随机变量为投掷一个骰子结果的和，计算该随机变量的方差。

5.2.02 设随机变量为投掷两个骰子结果的和，计算该随机变量的方差。

5.2.03 设随机变量为投掷三个骰子结果的和，计算该随机变量的方差。

5.2.04(*) 设随机变量为投掷 n 个骰子结果的和，计算该随机变量的方差。

5.2.05 设一枚硬币正面朝上的概率为 p，X 是随机变量，表示出现两次正面朝上时所投掷的次数，求 X 的方差。

5.3 车贝雪夫不等式

计算一个随机变量 X 的方差 $\sigma^2(X)$ 的意思就是要说明随机变量 X 是如何偏离其期望值的。车贝雪夫（Chebycheff）不等式给出了这个问题的实质，它也是**大数定律**的早期稍弱一点的形式。

定理 设随机变量 X 的期望值为 μ，方差为 σ^2，令 $t>1$，则有

$$P(|X-\mu|\geq t\cdot\sigma)\leq\frac{1}{t^2}$$

证明 使用刚才定义的 σ^2，即

$$\sigma^2=E(|X-\mu|^2)=\sum_x P(X=x)\cdot(x-\mu)^2\geq\sum_{x:\ |x-\mu|\geq t\sigma}P(X=x)\cdot(x-\mu)^2$$

这是因为在可能出现的较小的指标集上计算非负数的和，不会增加和的值。这也就是

$$\sum_{x:\ |x-\mu|\geq t\sigma}P(X=x)\cdot(x-\mu)^2\geq\sum_{x:\ |x-\mu|\geq t\sigma}P(X=x)\cdot(t\sigma)^2$$

这是因为在满足 $|x-\mu|\geq t\sigma$ 的实数 x 的集合上，用 $(t\sigma)^2$ 代替 $(x-\mu)^2$ 当然也不会增加和式的值。由此就有：

$$\sum_{x:\ |x-\mu|\geq t\sigma}P(X=x)\cdot(t\sigma)^2=(t\sigma)^2\sum_{x:\ |x-\mu|\geq t\sigma}P(X=x)$$

因为常数 $t\sigma$ 与指标 x 无关，于是

$$\sigma^2 \geqslant (t\sigma)^2 P(|X-\mu| \geqslant t\sigma)$$

重新整理并消去 σ^2 即得：

$$\frac{1}{t^2} \geqslant P(|X-\mu| \geqslant t\sigma)$$

定理得证。 ♣

习题

5.3.01 估计在抛掷 100 次硬币的试验中，正面朝上的次数在 $40 \leqslant x \leqslant 60$ 之间的概率。

5.3.02 估计在抛掷 1000 次硬币的试验中，正面朝上的次数在 $900 \leqslant x \leqslant 1100$ 之间的概率。

5.3.03 估计在抛掷 10000 次硬币的试验中，正面朝上的次数在 $9000 \leqslant x \leqslant 11000$ 之间的概率。

5.3.04 一个硬币出现正面朝上的概率为 1/10，证明在 100 次的抛掷中，硬币正面朝上的次数超过 20 次的概率小于 1/9。

5.3.05 一个硬币出现正面朝上的概率为 1/10，证明在 10000 次的抛掷中，硬币正面朝上的次数小于 2000 的概率小于 1/900。

5.4 大数定律

应用车贝雪夫不等式我们可以证明被称为（弱）**大数定律**的一种特殊情形，至少是在概率空间 $\{H,T\}$ 上重复试验的情形是可以证明的，这里 $P(H)=p$，$P(T)=1-p$。

将期望值作为随机变量的值的想法太幼稚，因为期望值只是一个平均值。但我们可以确切地知道我们得到的大多数值都会很接近于期望值，从数量表现上它还依赖于方差。有多种方法可以阐述这一结论，而且这一结论对于比我们这里所提到的随机变量还要广泛的随机变量也成立，认识到这一点是很重要的，我们只是选择了一个简单的例子以避免过多的不必要的技术细节。

定理 在概率空间 $\Omega=\{H,T\}$ 上，$P(H)=p$，随机变量 X 表示在 n 次试验中有多少次 H 发生，ε 表示任意小的正整数，则

$$\lim_{n\to\infty} P(|X-p\cdot n| > \varepsilon \cdot n) = 0$$

这里我们知道 X 的期望值为 $p\cdot n$。

证明 我们将利用车贝雪夫不等式，通过合理选择参数得出这个结论。设 μ 为 X 的数学期望值，σ 为其标准差。由前面的计算我们知道 $\mu=p\cdot n$，$\sigma=\sqrt{p(1-p)}\cdot\sqrt{n}$。由车贝雪夫不等式得到 $P(|X-\mu|>t\sigma)<\frac{1}{t^2}$，所以

$$P(|X-p\cdot n| > t\cdot\sqrt{p(1-p)}\cdot\sqrt{n}\ ) < \frac{1}{t^2}$$

取

$$t=\frac{\varepsilon}{\sqrt{p(1-p)}}\cdot\sqrt{n}$$

得到

$$P(|X-p\cdot n|>\varepsilon\cdot n)<\frac{\sqrt{p(1-p)}}{n}$$

当n趋向于无穷，不等式右边也趋近于0，这就证明了定理。 ♣

习题

5.4.01 需要抛掷多少次硬币，才能使得

$$\frac{1}{2}\leqslant\frac{\text{正面朝上的次数}}{\text{反面朝上的次数}}\leqslant 2$$

的概率至少为9/10?

5.4.02 需要将一对骰子掷多少次，才能使得出现一次点数为7的概率至少为3/4 ？

5.4.03 需要将一对骰子掷多少次，才能使得出现一次点数为12的概率至少为9/10 ？

5.4.04 需要将一对骰子掷多少次，才能使得出现一次点数为7的概率至少为9/10 ？

第6章 现代对称密码

6.1 设计目标

通过分析一些密码的例子及它们失败的原因，我们能够更准确地确立一个成功密码的设计目标。首先，回顾一下迄今为止我们所见到的例子。

对于一个**简单代替密码**或**简单置换密码**而言，其密钥是一个对字母表 $\{a,b,c,\cdots,z\}$（或任何可用到的字母表）的**置换** f，加密步骤表示如下：

$$E_f(x_0,x_1,x_2,\cdots)=(f(x_0),f(x_1),f(x_2),\cdots)$$

这也称为**单表**代替密码，其特点是无论该字母出现在明文的什么位置，都对它做相同的代替变换。

集合 $\{a,b,c,\cdots,z\}$ 的置换函数 f 是一个简单的双射函数：

$$f:\{a,b,c,d,\cdots,y,z\}\rightarrow\{a,b,c,d,\cdots,y,z\}$$

因此，我们可以把 f 看作是对字母表的“混合处理”。每一个步骤都是用了相同的代替 f。解密步骤可以使用反函数 f^{-1} 代替 f：

$$D_f=E_{f^{-1}}$$

因此：

$$D_f(x_0,x_1,x_2,\cdots)=(f^{-1}(x_0),f^{-1}(x_1),f^{-1}(x_2),\cdots)$$

所有的*移位*和*仿射*密码都是简单的代替密码的例子。“密语”是此类密码最常见的形式。在移位和仿射密码中，置换是非常严格的一类置换，我们将以模 26 的数学方法描述。

通过使用单字母频率和其他关于英语（或其他正在使用的自然语言）结构的基本信息，针对单表代替密码的唯密文攻击是成功的。这是非常糟糕的。

为了增加复杂性，采用某些方法，在不同的步骤中尽可能的使用不同的代替方法，这就是**多表代换密码**。对于**周期性多表代换密码**的常见描述为：密钥是代替变换 f_i 的 m 维数组 $(f_0,f_1,\cdots,f_{m-1})$（其中 m 为密钥的一部分），加密算法为：

$$E_f(x_0,x_1,x_2,\cdots)=(f_0(x_0),f_1(x_1),\cdots,f_{i\%m}(x_i),\cdots)$$

其中，$f_{i\%m}$（下标为 i 模 m 的余数）用于加密明文的第 i 个字符。解密算法为：

$$D_f(x_0,x_1,x_2,\cdots)=(f_0^{-1}(x_0),f_1^{-1}(x_1),\cdots,f_{i\%m}^{-1}(x_i),\cdots)$$

因此，对明文中第 i 个字符的加密仅依赖于这个字母以及 i 模 m 的值。所谓**周期**是指每隔 m 个字符，单个字母的加密就会重复。维吉尼亚密码是代替密码中周期性的多表代替密码。

周期性多表代替密码（如维吉尼亚）失败的本质原因就是攻击者能够将密文序列分解成若干由单表代替密码加密的*子序列*。

通过对比，使用不同的机制，一个**(简单的)换位密码**不改变单个字母，而是改变他们在消息中的位置。一些比较独特的古典密码的例子包括通过格子或将写了信息的纸带缠绕在圆

锥体上来阅读信息，都是这一类的密码。概要描述为：选择一个大小为 m 的正整数块，f 为 $\{0,1,\cdots,n-1\}$ 到它自身（集合上的一个置换）的映射函数，m 和 f 为密钥，对明文

$$x=(x_0,x_1,\cdots)$$

加密时，将其分割成长度正好为 m 的片段

$$(x_0,\cdots,x_{m-1})$$
$$(x_m,\cdots,x_{2m-1})$$
$$(x_{2m},\cdots,x_{3m-1})$$
$$\cdots$$

按如下方式加密每一个**分块** $(x_{km},\cdots,x_{km+m-1})$，为了描述简便，我们将一个固定的分块 $(x_{km},\cdots,x_{km+m-1})$ 表示为：

$$z=(z_0,\cdots,z_{m-1})=(x_{km},\cdots,x_{km+m-1})$$

于是

$$E_{m,f}(z)=(z_{f(0)},z_{f(1)}z_{f(2)},\cdots,z_{f(m-1)})$$

实际上这是通过下标的置换来描述的。这样做的效果就是移位仅对分块内的字符进行，没有对字母自身进行改动（除它们的位置之外）。常见的被称为变位字的**字谜**（文字游戏）就是通过换位密码来加密的。对于这类密码而言，由于一些小的字谜（文字游戏）已经被人们当作消遣而破译，从而对其安全性造成了一些负面影响。

相对于代替密码而言，换位密码的复杂之处在于如何描述明文字母的移动以及这些移动变化是如何依赖于密钥的。古典式独特的迷人游戏并不能处理好换位与密码的关系，因为通常一旦密钥暴露，则没有可以替换的：密钥包含一个单片秘密（monolithic secret），因此不存在可替换的机制。

我们可以看到双重变位（double anagramming）是对简单换位密码的有效攻击。

无论是代替密码还是换位密码，它们的密钥都是一个置换。对于代替密码而言，字母本身的序列改变了，故消息中的每一个字符都变了，而对于换位密码而言，消息中字符的位置改变了，但每一个字符本身并没有改变。

一个对明文分组加密的密码如换位密码称为**分组密码**，与之相对应的**流密码**是对明文中单个字符的操作。因而，代替密码可以称为**流密码**（最简单的一种），因为对每一个字符的加密与明文字符前后无关。代替密码的周期性特点，如在维吉尼亚密码中所表现的那样，是一个致命的缺陷。

然而，在分组密码和流密码之间的界限并不是很清晰的，并且从相对抽象的观点看，这种界限也几乎是不存在的。毕竟，一些分组密码可以看作周期长度等于分组长度的流密码。

必须注意到的是一些传统密码不是代替密码就是换位密码，现代密码则是两者的有效结合。试图区分分组密码和流密码是同样无用的，因为实际上一些密码算法将明文处理为序列或分组序列，对每一个分组的加密依赖于它在分组序列中的位置。

在序列密码的上下文中，也可以把明文序列认为是拆开了的分组，加密一个分组可能依赖也可能不依赖于它前面的明文。如果对每一个分组的加密与它前面的明文无关，则该密码是**同步密码**（显然在军事上也称为密钥自动密钥）。如果对一个分组的加密与她前面的明文有关，则该密码是**异步密码**。

20 世纪 40 年代末，克劳伍德·香农（Claude Shannon） 就提出了一些很好的针对（对称）密码的设计准则。首先，他重申了 Kerckhoff 惟一需要保密的是密钥原理而不是大量的密码机理的原则。其次，香农强调一个好的密码将融合混淆和扩散。这不但听起来很吸引人，而且它准确的指出了传统密码的弱点。通过**混淆**的方法，密码可以对攻击者隐藏一些语言的局部特征。通过扩散密码可以混淆明文的不同部分，因而，没有什么还留在它原来的位置。对此的一个粗略解释就是无论是在一个小范围还是大范围内的结构都将被破坏。

例如，单表代替密码就不符合这些标准。如双字母（像“ee”）这样的局部特征在密文中将依然表现为双字母。并且单字母的出现频率将依然得到体现。

像维吉尼亚这样的多表代替密码在混淆（隐藏语言的局部特征）上是非常有效的，因为它不是在每一时刻都采用同样的方法加密同样的字符。但维吉尼亚密码在扩散上是失败的，因为它没有做任何的换位。该弱点（加上周期性的替代）将受到 Friedman 攻击。

从设计的角度讲，传统换位密码在扩散方面是很优秀的，因为它们所做的恰恰就是更换字母的位置。在理想情况下，它也的确能够消除小范围内的语言特征，从而达到混淆的目的。但是，如前面所评论的那样，描述怎样的混淆才能改变对密钥的依赖却是不那么容易。

但是，到现在为止，可以清楚地看到一个好的密码应该是同时使用代替和换位以达到香农的混淆和扩散的目标。许多当代（对称）密码重复循环以下的一个或几个步骤：代替然后换位，然后代替，再换位等等。并且每一个实施的细节都依赖于密钥。

更进一步来说，当我们将字符编码成数字（无论是采用模 26 还是用 0 1 这样的字符串）时，我们就能更好地使用代替和换位。

习题

6.1.01 (*) 是否存在一种与 Kerckhoff 和香农准则相违背的好的加密算法，即惟一需要保密的是密钥而不是大量的密码机制？

6.1.02 (*) 为什么能够抵抗已知和自适应的明文攻击是非常必要的，且胜于仅仅抵抗唯密文攻击。

6.1.03 (*) 既然一次一密密码相当的安全，为什么不能用作缺省的通用密码呢？

6.2 数据加密标准

数据加密标准 DES，已诞生 20 多年了，由于它的密钥空间很小，现在已显得很落伍。美国电子阵线基金（Electronic Frontier Foundation）花了几年时间综合评估了可用于破解 DES 密码的机器费用和破译速度，于 1998 年用了大约 100 000 美金研究出了一台能在两天内计算出 DES 密钥的机器。然而，DES 已经存在了 20 年，并且没有发现隐藏的或有值得怀疑的弱点。并且看起来**三重 DES**（进行 3 次 DES 加密）依然是安全的。同时，DES 无论是在硬件上还是软件上都较容易实现。总而言之，在对付新的密码分析上，DES 的安全性比许多后出现的对称密码好一些。许多软件中都已嵌入了 DES，所以无论我们是否宣布废弃它，在未来的很多年内它都将会被使用。即使会被一个安全性较高的标准所代替，要面对不断增加的计算速度，DES 依然可以提供足够的安全性。

如你想像的那样，DES 比传统密码复杂。在这里不厌其烦地描述 DES 的部分原因是为了让你明白它并不容易理解。特别是与其沉溺于细枝末节，还不如去简单的了解这里到底有

多少细节。

DES 使用长度为 64 位的密钥，其中 8 位为错误校验位，因此实际上密钥只有 56 位（其中每个字节的第 8 位是奇偶校验位）。目前仍具有竞争力的**高级加密标准**（AES），其密钥长度为 128 位、192 位和 256 位。DES（分组密码）一次加密一个 64 位的明文分组，AES 则将加密 128 位的分组。

关于 DES 的问题，一直存在着政治干预。在 1970 年初，IBM 的一个组设计了它的前身 **Lucifer**，最终由 NSA 将其设计完成。NSA 减少了它的密钥长度并且修改了一些被称为 S 盒的东西。花了 20 年时间依然没有发现其所固有的弱点，一些人推断也许 NSA 并没有植入陷门。密钥的长度之小一直被怀疑，但那不是个秘密。

在这里我们考虑 DES 的描述。一个精细的设计说明需要花费大量的计算时间用于验证。除了暴力攻击之外，已经发明了两种“现代”的密码分析的攻击方法：**差分密码分析法**和**线性密码分析法**。简单来说，差分密码分析法就是系统地研究明文中的一个细小变化是如何影响密文的，线性密码分析法是通过线性函数去逼近加密结果。任何一种检测模式都能提高攻击者寻找密钥的效率，如果一个诀窍可以使**工作量**减少到千分之一，其效果并不显著，但如果能够将工作量减少到 10^9 分之一，则具有决定性的意义。

DES 进行 16 轮加密，这意味着它是一个简单过程的重复，每一个过程称为 **Feistel 网络**，我们将简短的描述它。人们已发现如果轮数少一些（如 12 轮密码），会使密码明显地比完整的 DES 易受攻击。

Feistel 网络的基本思想比较简单，找出一个是自身反函数的可逆函数。确定一个正整数 n，在 DES 的情况下可为 32。提供一个长度为 $2n$ 的字符串，把它们分成两组，左半部分 L 和右半部分 R。我们可以把 L 和 R 看作长度为 n 的向量，每一项的值为模 2 的余数。将 f 当成只要输入 n 位的数据就产生 n 位输出的函数。符合 Feistel 网络的 F_f 将 $2n$ 位的 L 和 R 作为输入，通过以下函数产生的 $2n$ 位的输出：

$$F_f(L,R) = (L \oplus f(R), R)$$

其中 $\oplus$ 表示向量加法（对应位），然后再模 2 约简。（如果你能明白它的意义的话，它与比特对的异或 XOR 相同。）因为某些原因，习惯把这种加法写为 $\oplus$，虽然我们仅在这一小节中遵循这一惯例。

例如，$n = 5$ 的向量模 2 加法为：

$$\begin{aligned}&(1,1,1,0,0) \oplus (1,0,1,1,1)\\ &= (1+1,1+0,1+1,0+1,0+1)\%2\\ &= (2,1,2,1,1)\%2\\ &= (0,1,0,1,1)\end{aligned}$$

Feistel 网络密钥的特点就是如果你两次采用了相同的 f，你就能恢复出最初的输入：

$$F_f(F_f(L,R)) = F_f(L \oplus f(R), R) = (L \oplus f(R) \oplus f(R), R) = (L, R)$$

因为当我们采用模 2 约简的时候，对于任意矢量 v

$$v \oplus v = (0,0,0,\ldots)$$

其依据是 $0+0=0$ 和 $1+1=0$。

因此无论 f 多么古怪，我们都不用担心它的可逆性和如何找到它的反函数，这一点很好，

因为我们就可以集中精力去选择那些能够产生大量**混淆和扩散**的方法。我们使用一些基于密钥的有几分窍门的函数 f，通过一些彼此间简单的混合来重复这一过程。这正是 DES 所做的。

在选择了特定的函数 f，依赖于某一密钥的情况下，DES 的每一个**轮**都是我们前面描述的 Feistel 网络的增强版。

开始时，64 位的密钥每 8 位一组进行移位并根据以下规则重新排列，则称为**密钥置换**：

57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

这将按照顺序从左上角开始，从左到右，从上向下来读。这些数之间有一定的规律：在每一行上的数是以 8 递减的，到 0 的时候又重新开始。这些符号的含义在于对密钥实施置换，即密钥的第 57 位将换到第 1 位的位置（因为 57 是列表中的第 1 个数），第 49 位将换到第 2 位的位置（因为 49 是列表中的第 2 个数），第 41 位将换到第 3 位的位置（因为 41 是列表中的第 3 个数），依次类推。这种方法比较简单。

该算法的一个重要部分是每一轮中都使用 56 位密钥中不同的 48 位**子密钥**（我们从起始的 64 位密钥中去掉每一个第 8 位）。如何使用这些密钥被称为是**密钥的时序安排**。首先，对于 DES 的每一轮，密钥被等分成两部分，每部分 28 位，根据轮数，每一个部分都将被循环左移 1 位或 2 位。每轮移动的位数为：

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

（不要问我为什么！）通过被称之为**压缩置换**的方法，56 位的密钥就映射成了 48 位的子密钥，这意味着它是不可逆的。每一轮的**压缩置换**的规则都是相同的。

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

也就是说，第 14 位换到了第 1 位的位置（因为 14 是列表中的第 1 个数），第 17 位换到了第 2 位的位置（因为 17 是列表中的第 2 个数），这看上去的确很神秘。于是通过压缩置换的处理，移动 1 位或 2 位的变换以及对初始密钥的置换，结果就是为每一个 16 轮产生一个 48 位的**子密钥**。我们将简单描述如何使用它。

正如前面所指出的那样，每一轮 DES 对文本处理都依据该轮子密钥（以及在哪一轮）。L_i 表示第 i 轮的左半部分，R_i 表示第 i 轮的右半部分。因而，从一轮计算下一轮的公式可以表示为：

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus f(R_{i-1}))$$

其中，f 依赖于第 i 个子密钥。注意除了做 Feistel 网络处理外，我们还将文本分成左右两半进行调换，这样就难以区分真正的左右部分。我们必须对函数 f 进行描述并且解释它是如何依赖于第 i 个子密钥的。

在每一个轮中，首先针对右半部分 R_{i-1} 做一个特定的（相当简单的）**扩展置换处理或 E 盒**处理。它接收 32 位的输入，产生 48 位的输出，通过以下规则给出（使用与前面相同的系

统）

32	1	2	3	4	5	4	5	6	7	8	9	8	9	10	11
12	13	12	13	14	15	16	17	16	17	18	19	20	21	20	21
22	23	24	25	24	25	26	27	28	29	28	29	30	31	32	1

注意这种模式并不是非常复杂。但是，由于文本中一位的变化会引起密文中多位的变化，这就意味着对最初明文中的很少位的改变将导致密文中很多位的改变，这将是一个**雪崩效应**。即使只改变一位，经过 DES 16 轮的计算，最终会使密文改变许多位。雪崩效应因此而得名。

在每一轮，从 E 盒输出的 48 位都将与该轮 48 位的子密钥（取决于具体在哪一轮）进行异或处理（即模 2 加）。

DES 最关键且最神秘的部分就是对 48 位子密钥和来自于 E 盒的 48 位输出使用了**代替盒**或者 **S 盒**，这也是它获得安全性的重要部分。DES 算法共有 8 个 S 盒，每一个 S 盒都是 6 位输入 4 位输出。48 位的输入数据被分成 8 个 6 位的分组并分别送入 8 个 S 盒。（前 6 位使用第一个 S 盒处理，第 2 个 6 位使用第二个 S 盒处理，依此类推。）这些输出再次揉捏在一起产生一个 32 位的总的输出。8 个 S 盒中的每一个都可以描述成一个 4 行 6 列的表。表中的每一项都是一个 4 位的数，意味着范围是 0~15，该数（用二进制表示）将是 S 盒的输出。S 盒的 6 位的输入指定的行和列如下。假设 6 位为 $b_1,b_2,\cdots,b_5,b_6$，于是：

$$\text{行} = 2\cdot b_1 + b_6$$

$$\text{列} = 8\cdot b_2 + 4\cdot b_3 + 2\cdot b_4 + b_5$$

其中行和列的下标开始于左上角且从 0 而不是 1 开始。例如，6 位的 111010 定义的行是 10（在二进制位串中的第 1 位和第 6 位），列为 1101（在二进制中，中间的 4 个位）。换句话说，我们应该查看第 2 行第 13 列，所有的情况都从 0 开始。

从第 1 个到第 8 个的 S 盒如下：

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9

10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

在每一个轮中，S 盒的 32 位的输出值首先进入 **P 盒**，即按照如下规则完成一个真正的置换。

16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

（符号的解释和前面一样）。

经过 16 轮后，**左半部分和右半部分的位置并未颠倒**。接着就要进行**最后的置换**处理。规则为（除了 S 盒以外，所有的解释与前面相同）：

40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

列上的规律是明显的。从这里的输出是密文。

幸运的是因为 Feistel 网络的性质，并且初置换和末置换的选择相对比较简单，采用严格一致的处理过程（使用相同的密钥）就可以**解密**。

显然 S 盒能够较好地抵抗差分密码分析。Bihan 和 Shamir 提出的差分密码分析于 1990 年才引起人们的关注。一些类似有固定 S 盒的密码，明显的存在一些可能受到自适应明文攻击的弱点。

不希望直接受到的一种攻击是线性密码分析，由 Mitsuru Matsui 于 1993 年公布于众。然而，DES 可以较好地抵抗这种攻击。

最后，看起来没有一种攻击比穷举密钥搜索更好。它是成功的。

备注 以上的描述仅涉及单一分组的加密。当对大量数据进行加密的时候，如果对数据的每一个分组分别进行加密将是一个错误。假如这里有许多数据，攻击者可进行**代码本攻击**，已知密文，通过出现频率，明密对照（cribs）以及已知的密文，即使没有密钥的信息，也可以对明文进行推断。为了避免这种攻击，应该以**分组链接**的方式进行加密，也就是说，每一个分组都影响它下一个分组的加密，同时，阻止代码本攻击。

备注 基于同样的原因，对一些会话（消息）使用相同的 DES 密钥也不是明智之举，因为这样就能够创建代码本。实际上，目前的应用情形是对每一个消息都使用一个“新”密钥，称为**会话密钥**。但存在一些明显的密钥管理问题：这些会话密钥如何协商？在这里当前的使用中，公钥的方法是很重要的：如 Diffie-Hellman 密钥交换（在后面将讨论）等方法允许会话密钥建立在一个保密且低成本的情况下。也许很难想像这将如何完成，并且这种可能存在的情况的确是公钥密码系统“不可思议”的地方。

习题

6.2.01(*) 因为 DES 中的 S 盒没有变量的控制，它们是静态的。在与 DES 类似的密码中使用静态 S 盒可能存在哪些优势及不利因素？

6.2.02(*) 使用被称为扩展置换的方法比“普通”置换方法好的原因是什么？

6.2.03(*) 设计一个好的图示范例。在该例中，改变 DES 输入的一位，通过连续的多轮后，最终在输出中影响其他一些位。

6.3 高级加密标准[⊖]

AES 就是**高级加密标准**（Advanced Encryption Standard），它在 2000 年初还处于激烈的评选中。这个未来的标准还没有完全指定，但在将来的一些应用中，不管怎么样它将取代 DES。但是 DES（以及其他一些老的对称加密密码）已经在很多老的软件中存在，对 DES 的分析仍将在一段时间内存在。

AES 第一轮竞争已经结束，第 2 轮候选者为：

- IBM 的 MARS 算法
- RSA 实验室的 RC6 算法
- Joan Daemen 和 Vincent Rijmen 的 Rijndael 算法
- Ross Anderson 、Eli Biham 和 Lars Knudsen 的 Serpent 算法
- Bruce Schneier、John Kelsey、Doug Whiting 、David Wagner 、Chris Hall、Niels Ferguson 的 Twofish 算法

新标准的部分规范要求是非常明确的：无论如何，AES 应当满足以下要求。AES 应该有一个大的分组长度，使用 128 位取代 DES 中 64 位的分组。AES 应该允许 128 位、192 位和

⊖ AES 评选活动已于 2000 年 10 月结束，最终确定 Rijndael 算法为 AES 的唯一算法。2001 年 11 月美国国家标准和技术研究所已经正式发布 AES 为联邦信息处理标准第 197 号（FIPS197）。——译者注

256 位的密钥。也可以假设它比三重 DES（也希望比 DES 快）快。NIST 美国国家标准和技术研究所也要求一些候选密码算法放弃它的专利权，如果该密码被选为新标准的话。

相对于 DES 使用静态 S 盒（无变化的）的情况，一些候选人提出了一个有趣的主意，那就是使用**动态的 S 盒**。也就是说，在 S 盒中，密钥的参数在某种程度上进行了改变。这存在着一些优点，但是也潜伏着一些缺点。一方面，如果 S 盒是动态的，那么攻击者就很难预先分析它们了。但另一方面，密码设计者也不容易系统地测试动态的 S 盒用以确定它们是否具有令人满意的特征。

静态的 S 盒与动态的 S 盒的问题派生出另一个问题，那就是 S 盒的可分析性。看起来 DES 中的 S 盒最初显著的优点是较简单的，那就是它能抵抗数十年的实验分析。另一方面，有一些可证明特性的 S 盒也许比仅有通过实验或统计验证的方法更好。但是对于是否以及如何达到这一点还不是很清楚。

也许在 2000 年中的某个时候，新的标准将从第 2 轮的竞争者中选出。

习题

6.3.01 给出至少两个为什么 AES 允许密钥长度可变的不同原因。

6.3.02 解释为什么基于安全的考虑而鼓励在 AES 中使用比 DES 大的分组。

6.3.03 与使用根据不同来源获得的各种其他密码算法相比较，为什么我们需要一个像 AES 那样的标准密码算法？

第7章 整　数

7.1 整除性

对于普通整数集合 $\mathbf{Z}$ 以及整数加、减、乘，与除（在存在的情况下，并不是所有整数 x 和 y 的商 x/y 都是整数）的运算，我们都有一个非常好的直观认识。在这里，我们将给出一些特别是关于**整除性**的术语，并包括了最直接的**素性测试**。

并不是所有的整数都能被另一个整数整除，我们给出一个定义：两个整数 d 和 n，如果 n/d 是整数，则称整数 d **整除** n（或者说 d 是 n 的一个**因子**）。这等价于存在另一个整数 k 使得 $n=kd$。作为等价的术语，我们也可以等价地说如果 d 整除 n，则 n 是 d 的**倍数**。

也就是说最好使用标准的方法来描述：如果存在整数 k 使得 $n=kd$ 则称 d 整除 n（比使用商 n/d 表达更好）。

如果 n 的因子 d 不是 $\pm n$ 或者 ± 1 的话，则称 d 是一个**真因子**。n 的倍数 N 是一个**真倍数**，如果它不是 $\pm n$。符号 $d \mid n$ 读作“d 整除 n”。注意任何 d 都可以整除 0，因为 $d \cdot 0 = 0$。换句话说，0 惟一整除的一个数就是它本身。

一个没有真因子的正整数被称为**素数**，那么它只能被它自己、它的负数以及 ± 1 整除，通常我们只关注正素数。

以下是一个最简单但却不是一个最有效的测试素数的方法。该方法具有这样的性质：如果一个整数不是素数，则采用这个方法可以找到它的最小因子 d，且 $d>1$。

命题　正整数 n 是素数当且仅当它不能被任何整数 d 整除，其中 $1<d\leqslant\sqrt{n}$。

证明　首先，如果 $d \mid n$ 并且 $2<d\leqslant\sqrt{n}$，那么整数 n/d 满足

$$\sqrt{n}\leqslant n/d<n/2$$

（这里我们只在实数范围内考虑不等式！）因此，两个因数 d 和 n/d 都不可能是 ± 1 或者 $\pm n$。因此，n 不是素数。

另一方面，如果 n 有一个真分解为 $n=d\cdot e$，其中 e 是两个因子中大的那个，那么：

$$d=n/e\leqslant n/d$$

所以 $d^2\leqslant n$，因而，$d\leqslant\sqrt{n}$。 ♣

如果能同时整除 m 和 n 的整数仅仅只有 ± 1，则称 m 和 n **互素**或**互质**的。如果 m 和 n 是**互素**的，我们也可以说 m 对 n 是素数。对于一个正整数 n，与其互素的且小于等于 n 的正整数的个数可以表示为 $\varphi(n)$，被称为**欧拉函数**。（通过反复试验的方法来计算 $\varphi(n)$ 并不是最好的方法，稍后我们将给出更好的方法。）

命题

- 如果 $a \mid b$ 并且 $b \mid c$，那么 $a \mid c$。
- 如果 $d \mid x$ 并且 $d \mid y$，那么对任意整数 a 和 b 有 $d \mid (ax+by)$。

证明　如果 $a \mid b$，那么必然存在整数 k 使得 $ak=b$。如果 $b \mid c$，那么必然存在整数 ℓ 使

得 $b\ell = c$。那么，在第二个等式中使用 ak 代替 b，我们就可以得到：

$$c = b\ell = (ak)\cdot\ell = a\cdot(k\ell)$$

所以 $a \mid c$。

如果 $d \mid x$，那么必然存在整数 m 使得 $dm = x$。如果 $d \mid y$，那么必然存在整数 n 使得 $dn = y$。那么，

$$ax + by = a(md) + b(nd) = (am + bn)\cdot d$$

所以，$ax + by$ 是 d 的倍数。 ♣

命题 n 和 N 是两个整数，并且 $n \mid N$，那么对于任意整数 x 有

$$(x\%N)\%n = x\%n$$

证明 设 $N = kn$，k 为整数，设 $x = Q\cdot N + R$，其中 $0 \leqslant R \lhd |N|$，R 称为 x 模 N 的余数，再设 $R = q\cdot n + r$，其中 $0 \leqslant r \lhd |n|$，r 称为 R 模 n 的余数。因而：

$$x = QN + R = Q(kn) + qn + r = (Qk + q)\cdot n + r$$

因此 x 模 n 的余数也为 r。 ♣

如果整数 d 能够整除任何一个整数 $n_1,\cdots,n_m$，则称 d 是整数 $n_1,\cdots,n_m$ 的**公因子**。如果整数 N 是任何一个整数 n_i 的倍数，则称 N 是整数集 $n_1,\cdots,n_m$ 的**公倍数**。

定理 m 和 n 为不同时为 0 的整数，在 m 和 n 的所有公因子中存在惟一的公因子 d，使得对 m 和 n 的任何其他公因子 e 都满足 $e \mid d$，其中 $d > 0$，则该公因子 d 被称为**最大公因子**或 m 和 n 的 **gcd**。两个整数 m 和 n（不同时为 0）的最大公因子是以 $xm + yn$ 表示的**最小正整数**，其中 $x, y \in \mathbf{Z}$。

备注 m 和 n 的最大公因子表示为 $gcd(m,n)$。如果两个整数的最大公约数为 1，则这两个整数是**互素**或称**互质的**。从而，如果它们是互素的则可以称 m 对于 n 是**素数**。定理给出了 gcd 的一个古怪但重要的特征。

证明 设 $D = x_0m + y_0n$ 为形如 $xm + yn$ 的最小正整数。首先，我们证明 m 和 n 的因子 d 能够整除 D。将 m 和 n 表示为 $m = m'd$，$n = n'd$，其中 $m', n' \in \mathbf{Z}$，因而：

$$D = x_0m + y_0n = x_0(m'd) + y_0(n'd) = (x_0m' + y_0n')\cdot d$$

这的确表明了 D 是 d 的倍数。

另一方面，利用除法算法可以将 m 表示为 $m = qD + r$，其中 $0 \leqslant r < D$，于是：

$$0 \leqslant r = m - qD = m - q(x_0m + y_0n) = (1 - qx_0)\cdot m + (-y_0)\cdot n$$

因此，r 也可以通过 x' 和 y' 表示为 $x'm + y'n$。因为 $r < D$，并且 D 是按这种方式表示的最小的正整数，因而必有 $r = 0$，所以 $D \mid m$。同理 $D \mid n$。 ♣

按照对偶的概念，用倍数代替因子，可以得到以下推论：

推论 m 和 n 为不同时为 0 的整数，在 m 和 n 的所有公倍数中存在惟一的公倍数 N，使得对 m 和 n 的任何其他公倍数 M 都满足 $N \mid M$，其中 $N > 0$，该公倍数 N 被称为**最小公倍数**或 m 和 n 的 **lcm**。

备注 如同我们先前注意到的，通过将数 m 和 n 分解为素数因子的乘积，我们就可以很容易地找到它们的最大公因子和最小公倍数：对每一个素数 p，整除 gcd 的 p 的方幂是整除 m 和整除 n 的 p 的最小方幂。因为对于每一个素数都是这样，所以我们就得到了最大公因子的素因子分解。例如：

$$gcd(2^3\cdot3^7\cdot5^2\cdot11^3, 2^1\cdot3^2\cdot5^3\cdot7^2\cdot11^2) = 2^1\cdot3^2\cdot5^2\cdot11^2$$

因为2^1是2的两个方幂中较小的那个，3^2是3的两个方幂中较小的那个，5^2是5的两个方幂中较小的那个，7^0是7的两个方幂较小的那个，11^2是11的两个方幂中较小的那个。在m和n的因式分解中取每一个素数的两个方幂中较大的那个，就可以得到最小公倍数。

然而，我们很快就会看到使用这种方法（通过整数的素因子分解式）来计算最大公因子和最小公倍数的效率是非常低的。

习题

7.1.01 找出60的所有因子，你如何确认你已经找出了所有的因子？

7.1.02 找出80的所有因子，你如何确认你已经找出了所有的因子？

7.1.03 找出90的所有因子，你如何确认你已经找出了所有的因子？

7.1.04 找出96的所有因子，你如何确认你已经找出了所有的因子？

7.1.05 对所有小于100的数，要么确定它们是素数，要么将它们分解成素因子乘积。

7.1.06 直接由整除性的定义证明，如果$d|m$，则$d|(-m)$。

7.1.07 直接从整除性的定义证明，如果$d|x$和$d|y$，则$d|(x+y)$与$d|(x-y)$成立。

7.1.08 不用计算，观察1331和14 641不是素数。

7.1.09 不用计算，观察10 510 100 501不是素数。

7.1.10 找出10 001的大于1的最小因子d。

7.1.11 找出12 344 321的大于1的最小因子d。

7.1.12 找出2、4、8、16、32、64、128的最小公倍数。

7.1.13 证明对于任意整数n，如果$d|n$且$d|(n+2)$，则$d|2$。

7.1.14 证明对于任意整数n，n与$n+1$一定是互素的。

7.1.15 证明对任意整数n，在n、$n+2$、$n+4$中只有一个可以被3整除。特别地，除了3、5、7以外，再没有三个素数是n、$n+2$、$n+4$模式的。

7.1.16 证明对任意整数n，n与n^2+1是互素的。

7.1.17(*) 证明对任意整数n，$16n^2+8n+1$与$16n^2-8n+1$的最大公因子是1。

7.1.18(*) 证明对任意两个整数m和n，它们的最小公倍数$lcm(m,n)$存在，而且可以通过公式$lcm(m,n)=m\cdot n/gcd(m,n)$计算得到。（注意：不要使用最大公倍数存在的假设来证明该公式。）

7.1.19(*) m和n是两个互为素数的正整数，证明$\varphi(mn)=\varphi(m)\cdot\varphi(n)$。其中，$\varphi(N)$是欧拉$\varphi$函数，该函数用于计算比$N$小，并且与$N$互素的正整数的个数。

7.1.20(**) 随机选择的两个正整数有多大的可能性互为素数。

7.2　因式惟一分解

我们能够证明将整数分解成素数因子的乘积，且该因式分解的方法是惟一的。也许从直观上看起来是正确的，因为我们已经通过小的整数证明了这个假设的正确性。它确实是正确的。但值得引起注意的是这件事如何去证明，特别是稍后我们将试图证明一些空想出来的数的因式分解是惟一的，那时候我们的直觉就不够用了。因为稍后我们就会注意到，通常“各种各样”的数都能惟一分解为素因子的观点是不正确的，这一点必须注意。

在此，我们给出欧拉φ函数$\varphi(n)$的计算公式，$\varphi(n)$的定义为：

$$\varphi(n)=\text{与 } n \text{ 互素且取值范围为 } 0\leqslant i\leqslant n \text{ 的整数 } i \text{ 的个数}$$

这个定义可以用于直接确定小整数 n 的 $\varphi(n)$ 值，用这样穷举攻击的方法是非常低效的，我们将给出一个相当好的公式。

我们也看到了很多试图将整数分解成素**因子乘积**的天真算法。为了在给定的范围内获得所有素数的列表，我们就会提及**埃拉托色尼的筛选法**，这是一种相当有效的完成该任务的方法。

另外，我们有机会回顾一些代数等式，它们可以为我们提供一些捷径来确定一个给定的数是否为素数，或者是否可以被分解成素数的乘积。

定理（因式惟一分解） 每一个整数都可以用本质惟一的方式表示为素数的、或正或负的乘积：

$$n=\pm p_1^{e_1}p_2^{e_2}\cdots p_m^{e_m}$$

其中，$p_1,\cdots,p_m$ 是互不相同的素数，所有的指数都是正整数。

备注 “本质惟一”的意思是指以不同顺序表示乘积不是真正的“不同”。使用“互不相同”是典型的数学用法：它表明“它们中的任意两个都不同”。公式中的 $\pm$ 是必要的，因为 n 可能是负数而素数本身是正数。当然，±1 的因式分解不包含素数。

推论 正整数 N 可以因式分解为：

$$N=p_1^{e_1}p_2^{e_2}\cdots p_n^{e_n}$$

其中，$p_1,\cdots,p_n$ 是互不相同的素数，指数 e_i 是非负的整数，因此，N 的欧拉 φ 函数值为：

$$\varphi(N)=(p_1-1)p_1^{e_1-1}(p_2-1)p_2^{e_2-1}\cdots(p_n-1)p_n^{e_n-1}$$

该定理的证明要依据如下关键的引理，该引理看起来是显而易见的，其实不然。

引理 设 p 是一个素数，假设 a 和 b 是整数，且 $p|(ab)$，那么 $p|a$ 和 $p|b$ 中至少成立一个。

证明（引理的证明） 如果 $p|a$，就已经得到引理的结论了。假设 p 不能够整除 a，那么最大公因子 $gcd(p,a)$ 不是 p，但这个最大的公因子也应该是 p 的因子，并且是正数。因为 p 是素数，它的正因子除了它本身，就是 1 了。因此 $gcd(p,a)=1$，从而存在的整数 x 和 y 使得 $xp+ya=1$。

因为 $p|(ab)$，我们可以得到 $ab=hp$，h 为整数。

$$b=b\cdot1=b\cdot(xp+ya)=bxp+yba=(bx+yh)\cdot p$$

这表明 b 是 p 的倍数。 ♣

推论（引理的推论） 如果一个素数 p 能够整除乘积 $a_1a_2\cdots a_n$，那么 p 至少能够整除其中的一个因子 a_i。

证明（推论的证明） 采用归纳法。在引理中已证明 $n=2$ 是成立的。假设 $p|(a_1\cdots a_n)$，而乘积 $(a_1\cdots a_n)$ 可以记为：

$$a_1\cdots a_n=(a_1\cdots a_{n-1})\cdot a_n$$

根据引理，p 要么能整除 a_n 要么能整除 $a_1a_2\cdots a_{n-1}$。如果能整除 a_n，那我们就已经证明了结论。如果不是，那么有 $p|(a_1\cdots a_{n-1})$。通过归纳法，这意味着 p 能够整除 $a_1,a_2,\cdots,a_{n-1}$ 中的一个因子。总而言之，我们总结出 p 能够整除乘积 $a_1a_2\cdots a_n$ 中的一个因子。 ♣

证明（定理的证明） 首先，我们证明对任意整数必然存在因式分解，然后再证明它是惟

一的。仅考虑正整数的因式分解就足够了，因为很明显的看到，n和$-n$的因式分解是相对应的。

假设存在一个大于 1 的整数n，它不能分解成素数的乘积。因而，要么n不是素数，要么$n=n$是一个因子为素数的因式分解。因此，假设n有一个真分解$n=xy$，其中$x,y>0$。因为这个因式分解是真分解，所以x和y必然严格小于n。从而，x和y都能够分解成素数的乘积，把它们的因式分解式合起来，就是n的因式分解。这同一些整数没有素因子分解式的假设相矛盾。

现在来证明惟一性，假设：

$$q_1^{e_1}\cdots q_m^{e_m}=N=p_1^{f_1}\cdots p_n^{f_n}$$

其中（不失一般性）：

$$q_1<q_2<\cdots<q_m$$

$$p_1<p_2<\cdots<p_m$$

都是素数，指数e_i和f_i都是正整数。我们必须证明$m=n$，且对于每一个i都有$q_i=p_i$、$e_i=f_i$。

因为q_1能够整除等式的左边，那么它也应该整除等式的右边。因而，根据前面引论的推论，q_1必然能够整除右边的某一个因子。假设q_1能够整除p_i，因为p_i是素数，故必有$q_1=p_i$。

我们申明$i=1$。当然，如果$i>1$，那么$p_1<p_i$。因为p_1整除等式的左边，所以它能够整除其中的某个q_j，因而$p_1=q_j$，我们会得到如下的不等式：

$$p_1=q_j\geqslant q_1=p_i>p_1$$

这是不可能的，故$q_1=p_1$。

更进一步，用$q_1=p_1$的e_1次方幂去除，我们可以推断出必有对应的指数$e_1=f_1$。

剩下的惟一性证明就是对N做归纳法。首先，1 有惟一的因式分解，称为 1 的乘积。无论如何，2 是一个素数，它的因式分解为 2 = 2。下面进行严格的归纳，假设所有的小于N的整数N'可以惟一的分解为素数的乘积（需要证明的是N也可以惟一的分解为素数的乘积）。

由如下等式

$$q_1^{e_1}\cdots q_m^{e_m}=N=p_1^{f_1}\cdots p_n^{f_n}$$

中消去相同的项$q_1^{e_1}=p_1^{f_1}$：

$$q_2^{e_2}\cdots q_m^{e_m}=\frac{N}{q_1^{e_1}}=p_2^{f_2}\cdots p_n^{f_n}$$

我们已经假设所有的指数e_i都是正整数，因而，$N/q_1^{e_1}<N$。通过归纳，我们知道$N/q_1^{e_1}$可以惟一的分解为素数的乘积，并且我们可以推断出剩下的因子也互相匹配。这就证明了因式分解的惟一性。♣

下面我们证明关于欧拉φ函数值的推论：

$$\varphi(N)=(p_1-1)p_1^{e_1-1}(p_2-1)p_2^{e_2-1}\cdots(p_n-1)p_n^{e_n-1}$$

其中，$n=p_1^{e_1}p_2^{e_2}\cdots p_n^{e_n}$，因子$p_i$是不同的素数，所有的指数都是正整数。这个推论的证明可通过计数的方法来完成：我们将对介于 0 到$N-1$之间的与N有公因子的整数x进行计数，并减去重复的计数。通过惟一的因式分解，如果x与N有一个公因子，则它就与N有一个素数

公因子。这样就存在 N/p_i 个能够被 p_i 整除的数，那么我们可以说在指定范围内与 N 没有公因子的数的个数为：

$$N-\frac{N}{p_1}-\frac{N}{p_2}-\cdots-\frac{N}{p_n}$$

然而，一般来说这并不正确：我们把可被两不同 p_i 整除的数计算了两次，所以我们应该在所有表达式 N/p_ip_j 中再加上被减去的数，其中，$i\neq j$。但是这样我们又多加了一些东西，所以还应减去表达式 $N/p_ip_jp_k$，其中 i,j,k 互不相同。因此：

$$\begin{aligned}\varphi(N)&=N-\sum_i\frac{N}{p_i}+\sum_{i\neq j}\frac{N}{p_ip_j}-\sum_{i,j,k\text{互不相同}}\frac{N}{p_ip_jp_k}+\cdots\\&=N\cdot(1-\frac{1}{p_1})(1-\frac{1}{p_2})\cdots(1-\frac{1}{p_n})\\&=p_1^{e_1}(1-\frac{1}{p_1})\cdot p_2^{e_2}(1-\frac{1}{p_2})\cdots p_n^{e_n}(1-\frac{1}{p_n})\\&=(p_1-1)p_1^{e_1-1}(p_2-1)p_2^{e_2-1}\cdots(p_n-1)p_n^{e_n-1}\end{aligned}$$

这正是我们所希望的公式。

备注 在证明的结尾部分我们用到的方法是一种解决计数问题的非常重要的原理，有些时候它被称为**容斥原理**（inclusion-exclusion principle）。

获得**素数因子分解**的最明显（但不是很有效）方法是上面提到的素性检验法。如下所示。通过用整数 $d=2,3,4,5,6,7,\cdots\leqslant\sqrt{N}$ 去除 N，直到找出 N 的最小的因子 d_1 为止，其中 $d_1>1$，或者判定 N 没有小于 $\sqrt{N}$ 的因子。在后一种情况，N 是素数。在前一种情况，通过用整数 $d=d_1,d_1+1,d_1+2,\cdots\leqslant\sqrt{N/d_1}$ 去除 N/d_1，直到找到 N/d_1 的大于 1 的最小因子 d_2，或者是确定 N/d_1 没有小于 $\sqrt{N/d_1}$ 的因子。在后一种情况，N/d_1 是素数。在前一种情况，通过用整数 $d=d_2,d_2+1,d_2+2,\cdots\leqslant\sqrt{N/d_1d_2}$ 去除 N/d_1d_2 来寻找它的最小因子 d_3，且 $d_3>1$，或者是判定 N/d_1d_2 没有小于 $\sqrt{N/d_1d_2}$ 的因子。在后一种情况，N/d_1d_2 是素数。在前一种情况下……

当 $N/d_1d_2\cdots d_m$ 为素数时，递归运算就结束了。同时，N 如果没有 $1<d<\sqrt{N}$ 范围内的因子 d，则 N 是素数。

备注 可用明显的方法稍加改进就能使上述程序更加简洁有效：如上所述，要求因子 d 时，没有必要用合数去做除法。因为如果 $d=ab$ 且 $a,b>1$，则我们早就能检测到 a 和 b 是不是一个因子。另一方面，被检验的数越大，则避免用一个合数去做试除法时就越加不利。

一些折衷的方法是可取的：没有必要对 2 以外的偶数去做试除法，也没有必要去试验 5 或 10 的倍数。该方法的依据时用十进制表示的整数，很容易判别是否可以被 2、5 或 10 整除。

提出一个略有不同的问题，我们如何找到小于 N 的所有素数。一个合理的寻找过程是**埃拉托色尼的筛选法**，描述如下。列出所有的从 2 到 N 的所有整数。

- 从 2+2 开始，标出列表中所有 2 的倍数。（就样就标出了列表中所有大于 2 的偶数。）
- 列表中还没有被标出来的下一个整数（在 2 的后面）是 3，从 3+3 开始，标出列表中所有 3 的倍数（那些已经标出的也计算在内）。（这样就标出了列表中所有大于 3 且为

3 倍数的数。）

- 列表中还没有被标出来的下一个整数（在 3 的后面）是 5，从 5+5 开始，标出列表中所有 5 的倍数（那些已经标出的也计算在内）。（这样就标出了列表中所有大于 5 且为 5 倍数的数。）
- ……
- 列表中还没有被标出来的下一个整数为 n，**n 为素数**，从 $n+n$ 开始，标出列表中所有 n 的倍数（那些已经标出的也计算在内）。（这样就标出了列表中所有大于 n 且为 n 倍数的数。）
- ……
- 直到你已经标出了小于 $\sqrt{N}$ 的最大素数的所有倍数，该过程就结束了。

例如，对 2 到 31 的一个整数列表，我们如果执行这一过程，我们首先制作列表：

2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21
22	23	24	25	26	27	28	29	30	31

标出列表中所有大于 2 的偶数

2	3	4*	5	6*	7	8*	9	10*	11
12*	13	14*	15	16*	17	18*	19	20*	21
22*	23	24*	25	26*	27	28*	29	30*	31

加标出列表中所有大于 3 且为 3 倍数的数

2	3	4*	5	6*	7	8*	9*	10*	11
12*	13	14*	15*	16*	17	18*	19	20*	21*
22*	23	24*	25	26*	27*	28*	29	30*	31

加标出列表中所有大于 5 且为 5 倍数的数

2	3	4*	5*	6*	7	8*	9*	10*	11
12*	13	14*	15*	16*	17	18*	19	20*	21*
22*	23	24*	25*	26*	27*	28*	29	30*	31

根据这一原则，下一个没有标出的整数是 7，但因为 7 大于 $\sqrt{31}$，所以在列表中没有被标出的整数是**素数**。

在分析一些特殊形式的整数和特殊多项式的因式分解时，一些**标准的恒等式**是很有用的：

$$x^2-y^2=(x-y)(x+y)$$
$$x^3-y^3=(x-y)(x^2+xy+y^2)$$
$$x^3+y^3=(x+y)(x^2-xy+y^2)$$
$$x^4-y^4=(x-y)(x^3+x^2y+xy^2+y^3)$$
$$x^5-y^5=(x-y)(x^4+x^3y+x^2y^2+xy^3+y^4)$$
$$x^5+y^5=(x+y)(x^4-x^3y+x^2y^2-xy^3+y^4)$$

如此类推。需要注意的是对于奇指数有两个恒等式，对于偶指数只有一个恒等式。

例如，我们也许会对整数 n 的 n^3-1 这种形式是否为素数感到好奇，我们可以这样做：

$$n^3-1=(n-1)\cdot(n^2+n+1)$$

因此，如果因子 $n-1$ 和 n^2+n+1 都介于 1 和 n^3-1 之间，那么 n^3-1 就有一个真的因式分解，所以 n^3-1 不是素数。实际上，很容易看出每一个因子都满足大于 1 并且小于 n^3-1。可以看到，当 $n=2$ 时，表达式 n^3-1 的值是 7，这是一个素数。因此，我们最好不要试图去证明这个表达式永远不为素数。

当 $n>2$ 时，很明显 $n-1>2-1=1$，这是一个比较。另一方面，同样对于 $n>2$ 会有：

$$n-1<2\cdot 2\cdot n-1<n\cdot n\cdot n-1$$

因此，当 $n>2$ 时，$0<n-1<n^3-1$。也就是说，对于 $n>2$，n^3-1 不可能是素数。

在历史上源于娱乐而在今天依然有现实意义的一类特殊的数是形如 2^n-1 的数。如果这样的数是素数，那么它们被称为**梅森素数**（Mersenne prime）。不知道是否有无限多的梅森素数存在。

另一类特殊形式的数是 2^m+1，如果这样的数是素数，那么它们被称为**费马素数**。不知道是否存在无限多的费马素数。很明显，费马曾明确地认为 $2^{2^n}+1$ 可能是素数，但 100 年后，这一点被欧拉证明是错误的。

习题

7.2.01 将 1028、2057 分解成素数的乘积。

7.2.02 不使用计算器，找出 111 111 111 111 111 的因子。

7.2.03 不使用计算器，找出 110 111 101 111 011 的因子。

7.2.04 不使用计算器，找出 101 010 101 010 101 的因子。

7.2.05 证明/观察仅根据十进制数的一位还不足以判断该数是否可以被 3 或 7 整除。

7.2.06 解释当 $n>2$ 时为什么 n^2-1 不是素数。

7.2.07 解释当 $n>1$ 时为什么 3^n-1 不可能是素数。

7.2.08 解释为什么 2^m+1 不可能是素数除非 m 是 2 的幂。

7.2.09 我们知道 x^2-y^2、x^3-y^3、x^3+y^3 都可以进行因式分解，但在高中很少对更难的代数式 x^4+4y^4 进行因式分解，其实 x^4+4y^4 可以用两个三项二次式的因式分解表示，求该因式的分解（提示：$x^4+4y^4=(x^4+4x^2y^2+4y^4)-4x^2y^2$）。

7.2.10 如果整数 n 大于 1，n^4+4 会是一个素数吗?

7.2.11 将 x^6-y^6 按两种不同的方式进行因式分解。

7.2.12(*)（欧几里得关于素数无穷多的证明）假设只存在有限多个素数 $p_1,p_2,\cdots,p_n$。考虑 $N=p_1p_2\cdots p_n+1$，证明没有一个 p_i 能够整除 N。由此推断，必然存在某个不在该列表中的其他素数，从而产生了矛盾。

7.3 欧几里得算法

欧几里得算法（Euclidean algorithm）是寻找两个整数 m 和 n 的最大公因子 d 一个非常重要并且不是那么显而易见的系统化过程，它的有效性也是很好的。

它也提供了一个很好的算法来寻找满足下列等式的整数 x,y：

$$xm + yn = d$$

（这个表达式在最大公因子的存在性证明中出现过。）而且，欧几里得算法还提供了最有效的过程来寻找模 m 的乘法逆元素。我们在下面可以看到，欧几里得算法的每一个步骤都是除法算法的一个例子。

欧几里得算法的一个重要方面是它避免了将整数分解为素数因子，同时这也是实现这个目标的一个合理的快速算法。这对于手工计算和机器计算都是一样的。

使用欧几里得算法将 x, y 的最大公约数表示为 $ax + by = gcd(x, y)$ 的形式（以及寻找 x 模 y 的乘法逆元素的应用），可以使用一种节约“存储空间”的形式。在本节的结尾，我们将对此进行解释。

下面我们将举例描述欧几里得算法。

对于两个整数 513,614 执行欧几里得算法：

$$614 - 1 \cdot 513 = 101 \text{（614 模 513 约简）}$$
$$513 - 5 \cdot 101 = 8 \text{（513 模 101 约简）}$$
$$101 - 12 \cdot 8 = 5 \text{（101 模 8 约简）}$$
$$8 - 1 \cdot 5 = 3 \text{（8 模 5 约简）}$$
$$5 - 1 \cdot 3 = 2 \text{（5 模 3 约简）}$$
$$3 - 1 \cdot 2 = 1 \text{（3 模 2 约简）}$$

注意，第一步是给定两个数中的大数模小数的约简。第二步是小数模第一步中的余数的约简。在每一步中，前一步中的“模数”为下一步的“被除数”，而前一步中的“余数”成为下一步的“模数”。

在这个例子中，因为我们最后得到了余数 1，我们知道 614 和 513 的最大公因子就为 1。也就是说，614 和 513 互素。当计算快要结束时，我们就知道得到的最大公因子为 1，但是我们还是将整个过程做完。

注意，使用欧几里得算法来寻找最大公因子并不需要求出素因子分解式。因为将整数分解为素数是一个非常耗费时间的工作，这是公认的。

下面给出另一个例子，我们来寻找 1024 和 888 的最大公因子：

$$1024 - 1 \cdot 888 = 136 \text{（1024 模 888 约简）}$$
$$888 - 6 \cdot 136 = 72 \text{（888 模 136 约简）}$$
$$136 - 1 \cdot 72 = 64 \text{（136 模 72 约简）}$$
$$72 - 1 \cdot 64 = 8 \text{（72 模 64 约简）}$$
$$64 - 8 \cdot 8 = 0 \text{（64 模 8 约简）}$$

在这个例子中，因为我们得到余数为 0，我们必须观察前一行的余数：8。结果 8 就是 1024 和 888 的最大公因子。

在这一点上，有必要给出欧几里得算法在最差情况下的步骤数目的粗略估算值。特别是，我们给出的估算值表明，采用这种方法来寻找最大公因子，比将整数分解为素数因子乘积并逐一比较所有因子的方法要快得多，而且优势随着整数值的增大而更加明显。

命题　欧几里得算法计算两个整数 $x > y$ 的最大公因子所必需的步骤数小于或者等于 $2 \cdot \log_2 y$。

证明　我们来看在算法执行过程中的一些连续步骤，这些步骤类似如下所示：

$$\cdots = y'$$
$$x' - q'y' = r'$$
$$y' - q''r' = r''$$

其中，$0 \leqslant r' < y'$ 且 $0 \leqslant r'' < r'$。我们特别指出满足 $r'' < y'/2$。也就是说，我们认为在算法的每两步中，余数至少以 1/2 的比例减少。（注意所有符号都表示非负整数。）

如果已经满足 $r' \leqslant y'/2$，则因为 $r'' < r'$，就可以得到我们想要的结果。另一方面，如果 $r' > y'/2$（但是仍然满足 $r'' < r'$），则很明显 q'' 必须为 1，且有：

$$r'' = y' - q''r' = y' - r' < r' - y'/2 = y'/2$$

这样，在算法的前两个步骤：$x - qy = r$ 和 $y - q'r = r'$ 中，r' 满足 $r' < \frac{1}{2}y$。因此，经过 $2n$ 步后，余数就小于：

$$\frac{1}{2} \cdot (\frac{1}{2})^{n-1} \cdot |y| = (\frac{1}{2})^n \cdot |y|$$

（第一个因数 1/2 是来自于头两步，然后每两步就又得到另一个因子 1/2）。因为当余数的值满足 $\leqslant 1$ 时，算法停止，所以当 n 达到某个值时，就有：

$$(\frac{1}{2})^n \cdot |y| \leqslant 1$$

经过整理得到：

$$2^n \geqslant |y|$$

或者

$$n \geqslant \log_2 |y|$$

也就是说，n（最差情况）是满足这个不等式的最小整数。实际的步骤数为 $2n$，因此有：

$$\text{所需步骤数} \leqslant 2\log_2 |x| \leqslant 2\log_2 |y|$$

这就证明了命题。 ♣

到此为止，我们只是明白了怎样找到 $gcd(x,y)$。如上所述，对于较小的数，通过将 x, y 分解因子为素数并比较因子的值，我们也许会觉得不是一件很可怕的事情。然而寻找整数 a, b 使

$$gcd(x,y) = ax + by$$

对小整数 x, y 也是一件很麻烦的事情。

欧几里德算法提供一种找 a, b 的方法只是有点麻烦，需要记下在欧几里德算法中出现的所有数字，然后进行回代，具体如下所示。

考虑并利用求 614 和 513 最大公因子的过程：

$$\begin{aligned}
1 &= 3 - 1\cdot 2 \text{（算法的最后一行）} \\
&= 3 - 1\cdot(5 - 1\cdot 3) \text{（根据前一行表达式替换 2）} \\
&= -1\cdot 5 + 2\cdot 3 \text{（对 5 和 3 的项进行整理）} \\
&= -1\cdot 5 + 2\cdot(8 - 1\cdot 5) \text{（根据前一行表达式替换 3）} \\
&= 2\cdot 8 - 3\cdot 5 \text{（对 5 和 8 的项进行整理）} \\
&= 2\cdot 8 - 3\cdot(101 - 12\cdot 8) \text{（根据前一行表达式替换 5）} \\
&= -3\cdot 101 + 38\cdot 8 \text{（对 101 和 8 的项进行整理）}
\end{aligned}$$

$$= -3\cdot 101 + 38\cdot(513 - 5\cdot 101)\text{（根据前一行表达式替换 8）}$$
$$= 38\cdot 513 - 193\cdot 101\text{（对 513 和 101 的项进行整理）}$$
$$= 38\cdot 513 - 193\cdot(614 - 513)\text{（根据前一行表达式替换 101）}$$
$$= 231\cdot 513 - 193\cdot 614\text{（对 614 和 513 的项进行整理）}$$

也就是说，我们实现了目标，我们得到：$1 = 231\cdot 513 - 193\cdot 614$。

为了成功地实现这个算法，必须明确哪些数只是系数，哪些数需要用来自于算法前面部分更复杂的表达式来替换，这一点很重要。因此，很有必要将其执行过程写成这种形式，即系数写在前面，需要替换的数放在后面。

习题

7.3.01　不用因式分解，求 $gcd(102\ 313, 103\ 927)$。

7.3.02　不用因式分解，求 $gcd(82\ 933, 145\ 393)$。

7.3.03　不用因式分解，求 $gcd(216\ 793, 256\ 027)$。

7.3.04　求 $gcd(1112, 1544)$，并将其表示为如下形式：$1112x + 1544y$，其中 x, y 为整数。

7.3.05　求 $gcd(117, 173)$，并将其表示为如下形式：$117x + 173y$，其中 x, y 为整数。

7.3.06　求 $gcd(10\ 201, 32\ 561)$，并将其表示为如下形式：$10\ 201x + 32\ 561y$，其中 x, y 为整数。

7.3.07　求 $gcd(12\ 345, 54\ 321)$，并将其表示为如下形式：$12\ 345x + 54\ 321y$，其中 x, y 为整数。

7.3.08　对于给定的整数 n，证明两个整数 $n^3 + n^2 + n + 1$ 和 $n^2 + n + 1$ 的最大公因子只能是 1。

7.3.09　对于给定的整数 n，证明两个整数

$$n^3 + n^2 + n + 1 \text{和} n^8 + n^7 + n^6 + n^5 + n^4 + n^3 + n^2 + n + 1$$

的最大公因子只能是 1。

7.4　乘法逆元

现在我们利用欧几里得算法来求模 m 的**乘法逆元**。首先，我们可以证明它们在一定条件下存在。

我们来回想一下，如果满足下列等式，则称 y 是 x 模 m 的乘法逆元：

$$x \cdot y \% m = 1$$

例如，3 是 2 模 5 的乘法逆元，因为：

$$2 \cdot 3 \% 5 = 6 \% 5 = 1$$

但是，8 也是 2 模 5 的乘法逆元，因为也满足：

$$2 \cdot 8 \% 5 = 16 \% 5 = 1$$

也就是说，可能存在几个不同的 2 模 5 的乘法逆元。我们现在解决这个疑问以及存在性问题。

命题　设 m 为一个非 0，非 ±1 的整数，x 为与 m 互素的一个整数，则 x 就有一个模 m 的乘法逆元。特别地，满足表达式 $ax + bm = 1$ 的任意整数 a 就是一个 x 模 m 的乘法逆元。另一方面，如果 x 有一个模 m 的乘法逆元，则 x 和 m 是互素的。

证明　在证明最大公因子的存在性时，我们已经证明两个整数 x, y 的最大公因子通常是

具有形式 $ax+bm$（a,b 为整数）的最小正整数。特别地，如果 x,m 互素，那么它们的最大公因子就是 1，因此存在整数 a,b 满足 $ax+bm=1$。也就是说，可以得到 $ax=-bm+1$。因此，我们用表达式 $qm+r$ 来表示 ax，其中 $0\leqslant r\lhd|m|$，因为 m 为非 0，非 ±1 整数。因此，ax 模 m 为 1，这正是期望的结果。

另一方面，假设 y 是 x 模 m 的乘法逆元，也就是满足：$xy\%m=1$。因此，对于某个整数 q，有：

$$xy=q\cdot m+1$$

设 d 为 x 和 m 的公因子，并设 $x=dx'$ 及 $m=dm'$。那么，重新整理得到：

$$1=xy-qm=dx'y-qdm'=d\cdot(x'y-qm')$$

也就是说，d 整除 1，这表明 $d=\pm1$，因此 x 与 m 惟一可能的公因子为 ±1，这正是我们所要证明的结论。 ♣

推论 m 是非 0 非 ±1 的整数，x 是一个整数。用欧几里德算法求出 x 和 m 的最大公因子，假如最大公因子为 1，通过对欧几里德算法进行反向操作得到表达式 $ax+bm=1$，从而得到 x 模 m 的乘法逆元。

现在再来讨论所有可能 x 模 m 逆元的特征。

命题 假如 y 是 x 模 m 的乘法逆元，对于某个整数 y'，如果 m 整除 $y-y'$，那么 y' 也是 x 模 m 的乘法逆元。相反地，假如 y,y' 都是 x 模 m 的乘法逆元，则 $y-y'$ 被 m 整除。

证明 假设对于某个整数 m，$y-y'=km$，设整数 q 满足 $xy=qm+1$，根据定义 $xy\%m=1$，那么

$$xy'=x(y-km)=xy-xkm=(qm+1)-xkm=1+m(q-xk)$$

也就是说 $xy'\%m=1$。

另一方面，假设 $xy\%m=1$ 和 $xy'\%m=1$，k,k' 是满足 $xy=mk+1,xy'=k'm+1$ 的整数，那么 $x(y-y')=(k-k')m$，由此 m 整除 $y-y'$，这个技巧后面还将出现。由于 x 和 m 互素，存在整数 a,b 满足 $ax+by=1$，那么

$$\begin{aligned}y-y'&=1\cdot(y-y')=(ax+bm)(y-y')=a(x(y-y'))+bm(y-y')\\&=a((k-k')m)+bm(y-y')\end{aligned}$$

这里用 $(k-k')m$ 代替了 $x(y-y')$，利用前面的等式可得到

$$y-y'=m\cdot(a(k-k')+b(y-y'))$$

由此即得 $y-y'$ 的确是 m 的倍数。 ♣

习题

7.4.01 用反复试验的方法，求 3 模 7 的乘法逆元。

7.4.02 用反复试验的方法，求 5 模 11 的乘法逆元。

7.4.03 用反复试验的方法，求 7 模 11 的乘法逆元。

7.4.04 用反复试验的方法，求 5 模 13 的乘法逆元。

7.4.05 用反复试验的方法，求 9 模 11 的乘法逆元。

7.4.06 求 n 模 $2n-1$ 的乘法逆元。

7.4.07 求 n 模 $2n+1$ 的乘法逆元。

7.4.08 求 n 模 $3n+1$ 的乘法逆元。

7.4.09 求 n 模 n^2+1 的乘法逆元

7.4.10 求 n 模 n^2-1 的乘法逆元

7.5 乘法逆元的计算

当我们运用欧几里得算法由后向前回推试图找出整数 a,b，使得 x,y 的最大公因子可表为 $ax+by$ 时，我们可以稍微运用一点技巧。首先我们回忆一下两个 2×2 矩阵的乘积，一个 2×2 的矩阵可表示为

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

两个这种矩阵的乘积定义为

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} A & B \\ C & D \end{pmatrix}=\begin{pmatrix} aA+bC & aB+bD \\ cA+dC & cB+dD \end{pmatrix}$$

假如你还不熟悉这种特殊的形式，我们稍微解释一下。一般地，为了计算乘积的第 i 行第 j 列元素，我们只需要用第一个矩阵的第 i 行和第二个矩阵的第 j 列元素。即

$$\begin{pmatrix} a & b \\ * & * \end{pmatrix}\begin{pmatrix} A & * \\ C & * \end{pmatrix}=\begin{pmatrix} aA+bC & * \\ * & * \end{pmatrix}$$

这里用*表示我们不知道或不关心的元素。其他三个式子分别为

$$\begin{pmatrix} a & b \\ * & * \end{pmatrix}\begin{pmatrix} * & B \\ * & D \end{pmatrix}=\begin{pmatrix} * & aB+bD \\ * & * \end{pmatrix}$$

$$\begin{pmatrix} * & * \\ c & d \end{pmatrix}\begin{pmatrix} A & * \\ C & * \end{pmatrix}=\begin{pmatrix} * & * \\ cA+dC & * \end{pmatrix}$$

$$\begin{pmatrix} * & * \\ c & d \end{pmatrix}\begin{pmatrix} * & B \\ * & D \end{pmatrix}=\begin{pmatrix} * & * \\ * & cB+dD \end{pmatrix}$$

我们也可以用其他类型的矩阵。我们来特别看一下 2×1 的矩阵，

$$\begin{pmatrix} x \\ y \end{pmatrix}$$

我们可以用一个 2×2 矩阵对它进行左乘，同样有类似的公式：

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}=\begin{pmatrix} ax+by \\ cx+dy \end{pmatrix}$$

这种乘法能够用来表示欧几里得算法中的运算步骤。假设我们要对两个正整数 x,y 运用欧几里得算法，设 q_1,r_1 是使得下式成立的整数：

$$x-q_1\cdot y=r_1$$

（$0\leqslant r_1<y$），则要做的下一步是用 y,r_1 分别代替 x,y，这可用如下式子表示：

$$\begin{pmatrix} y \\ r_1 \end{pmatrix}=\begin{pmatrix} 0 & 1 \\ 1 & -q_1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}$$

在欧几里得算法的每一步，按照这样的方法做下去，就可得到一对整数 x_i,y_i（其中 $x_0=x,\ y_0=y$），

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_{n+1} \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix}$$

这里的 q_{n+1} 是满足 $0 \leqslant x_n - q_{n+1} \cdot y_n \lhd |y_n|$ 的整数，对所有的下标 i，令

$$M_i = \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix}$$

因此欧几里得算法继续直到

$$M_n M_{n-1} M_{n-2} \cdots M_3 M_2 M_1 \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} gcd(x,y) \\ 0 \end{pmatrix}$$

假如矩阵的乘积为

$$M_n M_{n-1} M_{n-2} \cdots M_3 M_2 M_1 = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

则我们就得到了

$$ax + by = gcd(x,y)$$

因此，为了得到高效的欧几里得算法，我们只是需要一个简单的计算 $M_n \cdots M_1$ 的高效方法。

设 i 个 M_j' 的累积表示为

$$\begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} = M_i M_{i-1} M_{i-2} \cdots M_3 M_2 M_1$$

当然（在保证矩阵乘法结合律的情况下）

$$\begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} = M_i \begin{pmatrix} a_{i-1} & b_{i-1} \\ c_{i-1} & d_{i-1} \end{pmatrix}$$

在一开始我们有

$$\begin{pmatrix} a_0 & b_0 \\ c_0 & d_0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

因此，为了减少在欧几里得算法执行过程中的“存储器”，我们只需记六个量：

$$(x_i, y_i, a_i, b_i, c_i, d_i)$$

这里的 $x_0 = x, y_0 = y, a_0 = 1, b_0 = 0, c_0 = 0, d_0 = 1$，为了从第 i 个六元组得到第 $i+1$ 个六元组，我们需要计算 q_{i+1}，使得 $0 \leqslant x_i - q_{i+1} \cdot y_i \lhd |y_i|$，因此

$$(x_{i+1}, y_{i+1}, a_{i+1}, b_{i+1}, c_{i+1}, d_{i+1}) = (y_i, x_i - q_{i+1} y_i, c_i, d_i, a_i - q_{i+1} c_i, b_i - q_{i+1} d_i)$$

在第 n 步中当 $y_n = 0$ 时，计算过程结束。这时，

$$a_n x + b_n y = gcd(x,y)$$

备注 我们仍没有证明欧几里得算法如我们已声明的那般运行。

习题

7.5.01 求 21 模 25 的乘法逆元。

7.5.02 求 11 模 26 的乘法逆元。

7.5.03 求 12 模 29 的乘法逆元。

7.5.04 求 210 模 251 的乘法逆元。

7.5.05 求 19 模 24 的乘法逆元。

7.5.06 求 2101 模 2513 的乘法逆元。

7.5.07 求 1234 模 4321 的乘法逆元。

7.5.08 求 21017 模 25139 的乘法逆元。

7.6 等价关系

认为整数模 m 与模 m 约简是有关联的想法是很诱人的，但这是一个危险的陷阱。为了讨论更多的复杂密码（其他事情），我们认为稍微丰富的定义符号是必要的，除此之外没有别的原因。

等价关系（定义在下面）的概念是传统的相等概念的概括和扩充，在整个数学中随处可见。相关的**等价类**（定义在下面）也具有同等重要的作用。

我们的目的就是要使概念和记号更加精确，比如用“$x \sim y$”表示 x 和 y 有一些共同的特性。我们可以为它建立一般的模型，而不用担心它们具体的特性是什么。

回忆一下集合 S 到集合 T 的函数 f 的正式定义：我们认为 f 是一种规则，它将一个输入 $s \in S$ “计算”或“关联”一个输出 $f(s) \in T$，出于某些原因，这个说法是不合理的。

集合 S 到集合 T 的函数 f 的正式定义（可能不够直观）是笛卡尔积 $S \times T$ 的一个子集 G，它具有如下性质：

- 对于每一个 $s \in S$，刚好存在一个 $t \in T$，使得 $(s,t) \in G$。

将这个性质与如下通常的符号联系起来就有：

$$f(s) = t \qquad \text{如果 } (s,t) \in G$$

（例如：S 和 T 是真正的直线，G 是 f 的一个图）。

在比较正式的文章中，首先对于集合 S 上的关系 R，有一个原始的一般概念：集合 S 上的**关系** R 简单地说就是 $S \times S$ 的一个子集。如果有序对 (x,y) 在 $S \times S$ 子集 R 中，则可记为 xRy。

与函数的正式定义相比较，“关系”的定义可以很清楚地表明每个函数都是一个关系。但是大部分的关系不满足函数的条件。“关系”这种定义除了为进一步研究提供条件之外没多大意义。

集合 S 上的**等价关系** R 是一种特殊的关系，它满足

- **自反性**：对所有的 $x \in S$，则有 xRx；
- **对称性**：假若 xRy，那么 yRx；
- **传递性**：假若 xRy 和 yRz，那么 xRz；

等价关系基本的例子是普通数值相等，或集合相等，或我们熟悉的其他形式的相等。等价关系的一个通用表示方法为：

$$x \sim y$$

（是~而不是 **R**）。有时读作 x **等价于** y，明确指出是等价关系。

在集合 $\mathbf{R}^2$ 上的一个等价关系的简单例子可定义为：

$$(x,y) \sim (x',y') \quad \text{当且仅当} \quad x = x'$$

也就是说，在解析几何中，两个点等价当且仅当他们在同一垂线上。读者容易验证该例子满足等价关系的三个性质。

设~表示集合 S 上的等价关系，对 $x \in S$，包含 x 的~等价类 $\bar{x}$ 是 S 的子集

$$\bar{x} = \{x' \in S : x' \sim x\}$$

S 上~的**等价类集合**表示为 $S/\sim$（好像我们是正在讨论某种商）。每个元素 $z\in S$ 当然包含在某个等价类中，使得对所有 $s\in S$ 的等价类都有 $s\sim z$。

注意一般等价类 $\bar{x}$ 的等式 $\bar{x}=\bar{y}$，$\bar{y}$ 并不是表明诸如 $x=y$ 此类的东西，而由 $x=y$ 总可得 $\bar{x}=\bar{y}$，一般在 $\bar{x}$ 中不只是包含 x 本身，还有其他元素。

命题 ~表示集合 S 上的等价关系，假如两个等价类 $\bar{x},\bar{y}$ 有共同的元素 z，则 $\bar{x}=\bar{y}$。

证明 假设 $z\in\bar{x}\cap\bar{y}$，那么 $z\sim x, z\sim y$，因此对于任意一个 $x'\in\bar{x}$，我们有

$$x'\sim x\sim z\sim y$$

利用~的传递性得到 $x'\sim y$，因此对于每个 $x'\in\bar{x}$，它确实也在 $\bar{y}$ 中。也就是说，$\bar{x}\subset\bar{y}$。同理，颠倒 x,y 的位置，可得到 $\bar{y}\subset\bar{x}$，因此 $\bar{x}=\bar{y}$。 ♣

认识到我们倾向于用 x 的等价类 $\bar{x}$ 的记法来表示等价类是很重要的，但没必要这样做。因而说“集合 S 上的等价关系~的等价类 A”是合法的。

当然，给定 S 上的等价类 A，在集合 S 中找到 x 使得 $\bar{x}=A$ 是很方便的。x 是等价类的**代表**。子集 A 的任一个元素都是一个代表，因此我们不能认为一个等价类只有一个代表。

命题 ~表示集合 S 上的等价关系，则 S 上~的等价类是互不相交的集合，它们的并集是 S。

证明 等价类的并集是整个集合并不惊奇：给定 $x\in S$，x 当然属于等价类

$$\{y\in S: y\sim x\}$$

令 A 和 B 是两个等价类，$A\cap B\neq\varnothing$，因此 $A=B$（作为集合）。由于交集非空，有元素 $y\in A\cap B$，根据“等价类”的定义，对于所有的 $a\in A$，我们有 $a\sim y$，类似地，对于所有的 $b\in B$，我们有 $b\sim y$，根据传递性，$a\sim b$，这对于所有的 $a\in A$，$b\in B$ 都成立，因此 $A=B$（由于 A 和 B 是等价类）。 ♠

一个集合 S 的非空子集的并是整个集合 S，且它们是互不相交的，称为 S 的**划分**，前一个命题反过来也是对的。

命题 S 是一个集合，X 是集合 S 的所有子集的集合，且使得 X 是 S 的一个划分。用 $x\sim y$ 定义是集合 S 上的关系~，当且仅当存在 $T\in X$，使得 $x\in T, y\in T$。也就是说，$x\sim y$ 当且仅当它们属于 X 的同一元素 T 中，则~是等价关系，并且它的等价类是 X 的元素。

证明 由于 X 中集合的并是整个集合 S，每个元素 $x\in S$ 必定包含在某个 $T\in X$（注意 T 是 S 的子集）。因此，我们有自反性 $x\sim x$。假如 $x\sim y$，则存在 $T\in X$ 且包含 x 和 y，当然有 $y\sim x$，所以我们得到了对称性。

最后，X 中集合的互不相交性保证每个 $y\in S$ 只存在于 X 的一个集合中。对于 $y\in S$，令 T 表示来自于 X 且包含 y 的惟一集合。如果 $x\sim y$ 且 $y\sim z$,则必有 $x\in T$ 和 $z\in T$，因为 y 属于来自 X 的确定子集。因此 x,y 都在 T 中，所以 $x\sim z$，于是我们得到了传递性。关于验证等价类是 X 中的元素这一部分，留作练习。

习题

7.6.01 证明 {1,2,3}×{1,2,3} 的子集 {(1,1),(2,2),(3,3),(1,2),(2,1) 是在集合 {1,2,3} 上的等价关系。

7.6.02 证明实数域上定义为 xRy 当且仅当 $x\lesssim y$ 的关系 xRy 不是一个等价关系。

7.6.03 X 是 S 的非空子集的集合，所有非空子集的并是整个 S 且互不相交（因此 X 是一

个划分）。令~表示由这个划分定义的等价关系。证明等价类是 X 的元素。

7.6.04 集合 {1,2,3,4} 有多少个等价关系。

7.7 整数模 m

我们在进行算术运算时，希望计算的结果是模 m 约简的，那么现在我们就证明可以做模 m 的约简。这个看起来似是而非的事实（并且是正确的）如果要用“原始”的形式证明则会耗费大量的篇幅。

如果有两个不同的整数 x, y 相差非零整数 m 的某个倍数，则我们说 x **模** m **同余于** y，记作

$$x \equiv y \bmod m$$

以后任何这样的关系被叫做模 m 的**同余**，m 是**模数**。等价的说法有 $x \equiv y \bmod m$ 当且仅当 $m \mid (x-y)$。

举一个例子，由于 $5 \mid (18-3)$，所以 $3 \equiv 18 \bmod 5$，这只是整除性记法上的不同。这个符号（源于 200 年前的高斯）的意思，就是让我们用可比较的特性，把同余当作是等于的变形。同余具有类似等于的性质，但这有待证明，不过证明也不复杂。在给出这些性质的同时，我们也将引入一些相应的术语。

命题　对于固定的整数 m，模 m 同余是一个等价关系，即如同前面定义的那样。

- 自反性：对于任意的 x，总有 $x \equiv x \bmod m$；
- 对称性：如果 $x \equiv y \bmod m$ 成立，那么有 $y \equiv x \bmod m$；
- 传递性：如果 $x \equiv y \bmod m$ 且 $y \equiv z \bmod m$，那么 $x \equiv z \bmod m$；

证明　因为 $x-x=0$，所以 $m \mid 0$，自反性成立。如果 $m \mid (x-y)$，那么 $m \mid (y-x)$，即 $y-x=-(x-y)$，因此对称性成立。设 $m \mid (x-y)$ 且 $m \mid (y-z)$，那么有整数 k, ℓ 有 $mk = x-y$，$m\ell = y-z$，于是有

$$x-z = (x-y)+(y-z) = mk + m\ell = m(k+\ell)$$

因此传递性成立。

命题得证。 ♣

模 m **的整数**是关于等价关系——模 m 同余的整数等价类的集合，记为 $\mathbf{Z}/m$（或 $\mathbf{Z}_m$）。给定一个整数 x 和模数 m，x 模 m 的等价类

$$\{y \in \mathbf{Z} : y \equiv x \bmod m\}$$

通常记为 $\bar{x}$，被称为 x 模 m 的**同余类或剩余类**。有的时候，记号上的一横可以略去不写，即直接简单地记为 x 但需要保证通过上下文能够明白这表示 x 模 m，而不是整数 x。

给出一个例子，对于模数 12，我们有

$$\bar{0} = \overline{12} = \overline{-12} = \overline{2400}$$

$$\bar{7} = \overline{7} = \overline{-5} = \overline{2407}$$

$$\bar{1} = \overline{13} = \overline{-11} = \overline{2401}$$

或等价地，

$$0 \bmod 12 = 12 \bmod 12 = -12 \bmod 12 = 2400 \bmod 12$$
$$7 \bmod 12 = 7 \bmod 12 = -5 \bmod 12 = 2407 \bmod 12$$
$$1 \bmod 12 = 13 \bmod 12 = -11 \bmod 12 = 2401 \bmod 12$$

备注 有一个传统的模 m 等价类代表元的集合表示形式，即

$$\{\bar{0},\bar{1},\bar{2}\cdots\overline{m-2},\overline{m-1}\}$$

事实上，一些有问题的文章将整数模 m 定义为这样一个集合，但这容易理解整数模 m 到底是什么东西。我们应该对整数模 m 约简的集合（实际上是 $\{0,1,2,\ldots,m-1\}$!）和整数模 m 的集合加以区别，后者是整数模 m 等价类的集合，是一个更加抽象的东西。

很明显，如下表示是正确的（举例来说）：

$$\mathbf{Z}/3 = \{\bar{0},\bar{1},\bar{2}\}$$

当然这样的表示也是正确的：

$$\mathbf{Z}/3 = \{\overline{10},\overline{31},\overline{-1}\}$$

并且还有许多不同的方式描述整数模 3 的集合。

也就是说 $\mathbf{Z}/m$ 不是整数集合 $\{0,1,2,3,\ldots,m-1\}$，而是整数模 m 等价类的集合。集合 $\{0,1,2,3,\cdots,m-1\}$ 是模 m 约简的集合（对此还没有专门的记号）。我们来看如下命题：

命题 给定两个整数 x,x'，令 $x=qm+r$，$x'=q'm+r'$，q,q',r,r' 为整数且 $0\leqslant r\lhd m|$ 和 $0\leqslant r'\lhd m'|$，则 $x\equiv x' \bmod m$ 当且仅当 $r\equiv r' \bmod m$。

证明 如果 $x\equiv x' \bmod m$，则存在整数 k 使得 $x'=x+km$，那么

$$r'=x'-q'm=(x+km)-q'm=x+m\cdot(k-q')=qm+r+m(k-q')$$
$$=r+m\cdot(q+k-q')$$

这就证明了 $r\equiv r' \bmod m$。另一方面的证明类似。 ♣

同余除了是一个等价关系之外，它还有基本算术运算的加法、减法和乘法。

命题 对于固定的模数 m，如果 $x\equiv x'$，那么对所有 y

$$x+y\equiv x'+y \bmod m$$
$$xy\equiv x'y \bmod m$$

事实上，如果 $y\equiv y'$ 那么

$$x+y\equiv x'+y' \bmod m$$
$$x\cdot y\equiv x'\cdot y' \bmod m$$

证明 只需要证明更加普遍的结论就足够了。因为 $x'\equiv x \bmod m$，所以 $m|(x-x')$，那么存在整数 k 使得 $mk=x'-x$，即我们有 $x'=x+mk$。类似地，对于整数 ℓ 我们有 $y'=y+\ell m$。于是

$$x'+y'=(x+mk)+(y+m\ell)=x+y+m(k+\ell)$$

所以就有

$$x+y\equiv x'+y \bmod m$$

而

$$x'\cdot y'=(x+mk)\cdot(y+m\ell)=x\cdot y+xm\ell+mky+mk\cdot m\ell=x\cdot y+m\cdot(k+\ell+mk\ell)$$

所以 $x'y'\equiv xy \bmod m$。 ♣

作为上述命题的一个推论，同余直接继承了普通算数运算的一些性质，比如因为 $x=y$ 即意味着有 $x\equiv y \bmod m$：

- 分配律：$x(y+z)\equiv xy+xz \bmod m$
- 加法结合律：$(x+y)+z\equiv x+(y+z) \bmod m$
- 乘法结合律：$(xy)z\equiv x(yz) \bmod m$
- 1 的性质：$1\cdot x\equiv x\cdot 1\equiv x \bmod m$
- 0 的性质：$0+x\equiv x+0\equiv x \bmod m$

利用这些性质，我们就可以放心地做模 m 运算，而不至于产生混淆。作为一种符号，我们用

$$\mathbf{Z}/m$$

来表示**整数模** m，并且如果 $x\equiv y \bmod m$，我们就将两个整数 x,y 视为相同的。因此在 $\mathbf{Z}/m$ 中只有 m 个元素，在为它们正好就是模 m 的 m 个同余类（剩余类）。通常我们将 $0,1,2,\cdots,m-2,m-1$ 当作 $\mathbf{Z}/m$ 中的元素，但这却是不太合适的。

还有一些比较实用的结论，而且也是我们习以为常的东西：

- $m\equiv 0 \bmod m$ 且对任何整数 k，$km\equiv 0 \bmod m$；
- $x+(-x)\equiv 0 \bmod m$；
- $x\pm m\equiv x \bmod m$ 且对任何整数 k，$x\pm km\equiv x \bmod m$；

注意到所有的讨论中，我们一次只考查一个模数 m。

推论　对于固定的模数 m，在每个剩余类中恰好存在一个整数是模 m 的约简。因此 $x\equiv y \bmod m$ 当且仅当 x,y 模 m 有同样的约简，也就是说，$x\equiv y \bmod m$ 当且仅当 x,y 被 m 除时有相同的余数。

证明　对于固定的整数 x，利用除法运算，则存在惟一的 $0\leqslant r\leqslant|m|$ 和整数 q 使得 $x=qm+r$，因此 $x-r=qm$ 可以被 m 整除，所以 x 和 r 在同一个剩余类中。因为 r 是约简的，因此这就证明了对于每个剩余类至少存在一个约简的代表元。

另一方面，（验证约简算法的惟一性!）假设有 $x\equiv r'$，r' 的取值范围为 $0\leqslant r'<|m|$。如果 $0\leqslant r'<r$，那么

$$0<r'-r=(r'-x)-(r-x)$$

是 m 的倍数，而且我们有 $0<r'-r\leqslant r'<|m|$。但 m 的倍数不可能满足 >0 且 $\lhd|m|$，所以不可能有 $0\leqslant r<r'$。另一方面，假设 $0\leqslant r'<r$，前面讨论的对称性，同样可以得到一个矛盾。因此 $r=r'$，这就证明了惟一性。 ♣

推论　对于给定的模数 m 和整数 x,y，为简便起见我们用

$$x\%m$$

表示 x 模 m 的约简，则

$$(x+y)\%m=((x\%m)+(y\%m))\%m$$

$$(x\cdot y)\%m=((x\%m)\cdot(y\%m))\%m$$

证明　$x'=(x\%m)$ 的剩余类与 x 本身的剩余类相同。因此模 m，我们就有

$$((x\%m)+(y\%m))\%m\equiv(x\%m)+(y\%m)\equiv x+y$$

因为我们已经证明了如果 $x'\equiv x$ 且 $y'\equiv y$，则 $x'+y'\equiv x+y$。类似的 $x+y\equiv(x+y)\%m$，由此利用传递性，有

$$((x\%m)+(y\%m))\%m\equiv(x+y)\%m$$

上述推导过程对乘法同样成立。 ♣

至此，我们明显得到了这样的印象，同余的所有性质完全遵循了普通等式的性质以及初等算术的性质。

我们再回到模 m 的乘法逆元问题上来。即为了求得元素 a 模 m 的乘法逆元，我们需要求解如下关于 x 的方程

$$ax \equiv 1 \bmod m$$

这里的整数 a 是给定的。除非 $a=\pm 1$，方程 $ax=1$ 的解 $x = 1/a$ 不是整数。但这不是我们所要的。实际上回想一下，如果 $gcd(a,m)=1,$，则存在整数 x,y，使得

$$ax + ym = gcd(a,m) = 1$$

那么 $ax-1=ym$ 是 m 的一个倍数，它包含着方程

$$ax \equiv 1 \bmod m$$

除非 $a=\pm 1$，如果仅仅因为 $x = 1/a$ 不是一个整数，这个 x 不可能是 $1/a$。我们需要的是另外一些新的东西。

之前我们已经证明，a 有一个乘法逆元当且仅当 $gcd(a,m)=1$，而且欧几里得算法确实是求乘法逆元的有效方法。

根据前面讨论的结果，我们对与 m 互素且有乘法逆元的模 m 整数指定一个专门的符号：

$$\mathbf{Z}/m^{\times}$$

此式中上标不是一个“x”，而是一个“乘”的符号，是指乘法与乘法逆元。

命题 两个与 m 互素的整数 x,y 的乘积 xy 仍然是与 m 互素的。

证明 考虑这一问题的一种方法是利用整数的素因子分解，但我们不用它。我们利用如下事实：两个整数 a,b 的最大公因子可表示为

$$gcd(a,b) = sa + tb$$

其中 s,t 为整数。由已知条件，存在整数 a,b,c,d 使得

$$1 = ax + bm \qquad 1 = cy + dm$$

那么就有

$$1 = 1 \cdot 1 = (ax + bm)(cy + dm) = (ac)(xy) + (bcy + axd + bdm)m$$

1 可表示为 $A(xy) + Bm$ 的形式，因经 xy 和 m 是互素的。 ♣

所以在 $\mathbf{Z}/m^{\times}$ 中，我们就可以做乘法和求逆运算。

习题

7.7.01 求一个整数 x 使得 $(x\%1009)\%1013 \neq x\%1013$。

7.7.02 两个正整数 n 和 N，其中 n 不能整除 N，求一个整数 x 使得 $(x\%N)\%n \neq x\%n$。

7.7.03 集合 $\mathbf{Z}/n$ 有多少个元素？

7.7.04 集合 $\mathbf{Z}/30^{\times}$ 有多少个元素？

7.7.05 集合 $\mathbf{Z}/24^{\times}$ 有多少个元素？

7.7.06 用模 10 约简 100000000001。

7.7.07 先计算再用指定的模数约简：110×124 模 3，12+1234567890 分别模 10。

7.7.08 在 2^{99} 的十进制展开式中，计算它的个位数字。

7.7.09 计算 $2^{1000}\%11$（提示：不要用大于 11 的数）。

7.7.10 计算 $3^{1000}\%11$（提示：不要用大于 11 的数）。

7.7.11 在3^{999}的十进制展开式中，计算它的个位数字。

7.7.12 求 3 模 100 的乘法逆元。

7.7.13 求 1001 模 1234 的乘法逆元。

7.7.14 找出三个不同的模 105 的剩余类x，使得$x^2 \equiv 1 \bmod 105$。

7.7.15 找出三个不同的模 143 的剩余类x，使得$x^2 \equiv 1 \bmod 143$。

7.7.16 证明$x^2 - y^2 = 102$没有整数解。（提示：当考虑平方时，总是首先试验模 4，检验模 4 仅有的平方是 0 和 1）

7.7.17 证明$x^3 + y^3 = 3$没有整数解。（提示：考察模 7 的立方是多少？）

7.7.18 证明$x^3 + y^3 + z^3 = 4$没有整数解。（提示：模 9？）

7.7.19 证明$x^2 + 3y^2 + 6z^3 - 9w^5 = 2$没有整数解。

7.7.20 舍 9 法：证明对于一个整数$n = a_k \cdots a_1 a_0$，有$n \equiv a_k + \cdots + a_1 + a_0 \bmod 9$，其中$a_k, \cdots, a_1, a_0$是十进制数。

7.7.21 用舍 9 法证明 123456789123456789+234567891234567891≠358025680358025680。

7.7.22 用舍 9 法证明

$$123456789123456789 \times 234567891234567891$$
$$\neq 28958998683279996179682996625361999$$

当然不能直接用手工方法检查，也许大多数计算器将会发生溢出。

7.8　本原根和离散对数

在这一节中我们将简单解释什么是本原根（primitive root），并阐述我们所做假设的正确性。本原根的存在性将在以后证明，这个证明需要做大量的准备工作。本节还将定义离散对数。

设n为正整数，整数g是**模**n的一个**本原根**,如果对于每个与n互素的x都存在一个整数l，使得

$$g^{\ell} = x \bmod n$$

对于给定的g和给定的x,具备这一特性的整数ℓ则是x的以g为底模n的**离散对数**。为了避免于其他普通的对数相混淆，有时x的以g为底模n的离散对数被称为x的以g为底模n的**指数**。

对于大多数整数n，不存在模n的本原根。作为一个例子，对于模素数 7 的情形，3 是一个本原根，因为（均模 7）

$$3^1 = 3$$
$$3^2 = 9 \ = 2$$
$$3^3 = 27 = 6$$
$$3^4 = 81 = 4$$
$$3^5 = 243 = 5$$
$$3^6 = 729 = 1$$

这里通过求连续的方幂，我们看到所有非 0 的数模 7 都是 3 的方幂（当然是模 7）。这种方法被称作穷举验证。

相比而言，整数模 8 得到 1,3,5,7，但所有这些都不是本原根。通过计算我们看到 1 的幂

是 1，3 的幂模 8 是 3 或 1，5 的方幂模 8 是 5 或 1，7 的方幂模 8 是 7 或 1。模 8 没有本原根。

目前存在更多我们无法使用词汇进行解释的结构。稍微做一些解释，至少精确说明，模 m 的本原根什么条件下存在。如定理所述：

定理(后面证明) 对于那些存在模 n 本原根的整数 n，它们具有如下形式：

- $n=p^e$ 有奇素数 p，并且 $e\geq 1$；
- $n=2p^e$ 有奇素数 p，并且 $e\geq 1$；
- $n=2,4$；

特别地，存在模素数的本原根。

进一步，$\mathbf{Z}/n$ 是一个有限集，我们将预期它的元素列表，$h,h^2,h^3,h^4\cdots$（$\mathbf{Z}/n$ 的给定元素 h 的方幂）。虽然其本身无限，不会包含无限多个模 n 的不同元素。特别地，不管 h 是否为模 n 的本原根，我们可以想像存在正整数 t 满足：

$$h^t=1 \bmod n$$

满足上式的最小正整数 t 是 h（模 n）的**指数或阶**。后面我们可以看到在一些特殊情况下的本原根是存在的，我们还将证明：

命题 设 p 是一个奇素数，e 是一个正整数。设 g 是模 p^e 的一个本原根，则

$$g^{p-1}=1 \bmod p$$

更一般地

$$g^{(p-1)p^{e-1}}=1 \bmod p^e$$

这是具备这些性质的最小正指数。也就是说，模一个素数 p 的本原根的阶是 $p-1$，模 p^e 的本原根的阶是 $(p-1)p^{e-1}$(后面证明)。

备注 在给出这些结论的证明这前介绍它们，是因为在我们的应用中要用到它们，而其这些证明的某些方面是很难懂的。后面的章节的内容将直接关系到本原根、阶、指数和离散对数问题：费马小定理、欧拉定理、分圆多项式和其他的内容。

习题

7.8.01 验证 2 是模 5 的一个本原根。

7.8.02 验证 2 是模 11 的一个本原根。

7.8.03 验证 2 是模 25 的一个本原根。

7.8.04 验证 2 是模 10 的一个本原根。

7.8.05 求以 2 为底模 5 的 3 的离散对数。

7.8.06 求以 3 为底模 7 的 2 的离散对数。

7.8.07 求以 3 为底模 5 的 2 的离散对数。

7.8.08 求以 3 为底模 23 的 2 的离散对数。

7.8.09 求以 2 为底模 25 的 3 的离散对数。

7.8.10 求以 3 为底模 101 的 2 的离散对数。

7.8.11 求以 2 为底模 125 的 3 的离散对数。

7.8.12 选模 17 的两个不同本原根 r 和 s，$\log_r$ 和 $\log_s$ 分别表示以 r 和 s 为底的离散对数。验证 $\log_r 2 = \log_r s \cdot \log_s 2$。

7.8.13 证明，对于模 m 的两个不同本原根 r 和 s，$\log_r$ 和 $\log_s$ 分别表示以 r 和 s 为底的离散对数，则对任意模 m 非零的 x 都有

$$\log_r x = \log_r s \cdot \log_s x$$

第8章 希尔密码

按照现密码的标准衡量，希尔密码又是一个失败的例子，但是它揭示了一些比较有趣的东西。它是第一个分组密码之一，意味着它能够处理比单个字符大的多的信息块。

8.1 希尔密码原理

在了解了代替换密码的缺陷后（特别是周期密码），例如维吉尼亚密码，人们希望寻找一种不同的或者至少采用更加复杂机制的加密方法。

尽管分组密码与序列密码之间的区别通常仅仅只是一个度的问题，但是**希尔密码**（Lester Hill，1929）还是可以被看作一种分组密码。这种密码算法在机制上增加的数学特性似乎在当时给人们留下了深刻的印象，并从此使得数学在密码学中担任更重要的角色。在第二次世界大战中，该密码被用来对无线电呼叫信号进行加密。加密和解密均通过机械设备（而不是电子设备）来完成。

给定分组长度为 N（这也可以是密钥的一部分），并选择一个 $N \times N$ 可逆矩阵 K，其元素属于 $\mathbf{Z}/26$。此 K 即为密钥。将一个具有 N 个字符的分组 $y=(y_1, y_2, \ldots, y_N)$ 重新写为 $N \times 1$ 矩阵（列向量）：

$$y=\begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_N \end{pmatrix}$$

用矩阵乘法对此矩阵进行加密：

$$E_K(y)=K \cdot y$$

也就是说，$E_K(y)$ 中的第 l 个字符 $E_K(y)_l$ 为：

$$E_K(y_l)=\sum_{j=1}^{N} K_{lj} y_j$$

其中，K_{lj} 为 K 的第 (l, j) 个元素。这里我们可使用整数模 26 对字符进行方便的编码。

根据前面的假设，解密阶段 D_K 使用 K 的（乘法）逆矩阵 K^{-1}。实际上，根据矩阵乘法的结合律，使用密钥 K 进行解密与使用 K^{-1} 进行加密是一样的：

$$D_K(E_K(y))=K^{-1}(Ky)=(K^{-1}K)y=1_N y=y$$

注意，这并不是在寻找乘法逆矩阵，因为寻找乘法逆矩阵是很困难的，从而防范了非授权的解密行为。正如求大矩阵的逆矩阵一样，这个求解困难并不是主要的“秘密”。

这里的密钥空间潜在是无限大的，和维吉尼亚密码算法一样。因为从原理上看，对于明文数据块的长度 N（也就是矩阵的大小）是没有限制的。维吉尼亚密码算法当分块大小为 N

时，具有 26^N 个密钥。并不是每一个系数属于 $\mathbf{Z}/26$ 的 $N \times N$ 矩阵都是可逆的，只有超过 $\frac{1}{4}$ 的矩阵是可逆的（我们将在后面了解这一点）。因此，对分块大小为 N 的希尔密码，它共有超过 $\frac{1}{4} \cdot 26^{N^2}$ 个密钥。作为一种“纯粹的”分组密钥算法，这种密码算法当然是周期的，其周期等于所用的矩阵的大小。

需要注意的是，这既不是一种纯换位密码，也不是一种纯代替密码。因此，它没有那些纯形式的缺点。事实上，许多或大多数“现代”密码算法都是将换位密码和代替密码算法有机地结合起来的。但是希尔密码算法具有**线性**的缺点，我们将在后面进行讨论。

习题

8.1.01 请使用分组长度为 2 的希尔密码对“mydoghasbigfleas”进行加密，密钥为：$\begin{pmatrix} 3 & 17 \\ 1 & 6 \end{pmatrix}$。

8.1.02 已知希尔密码的密钥为 $K = \begin{pmatrix} 3 & 17 \\ 1 & 6 \end{pmatrix}$，一组用该密钥进行加密得到的密文为“XZIIAUCR”，求其明文。

8.2 对希尔密码的攻击

对希尔密码进行选择明文攻击并不困难。如果已知分组长度为 N，则加密 N 个字符串（每一个字符串长度为 N）：

$$
\begin{array}{c}
(1,0,0,\ldots,0,0) \\
(0,1,0,\ldots,0,0) \\
(0,0,1,\ldots,0,0) \\
\cdots \\
(0,0,0,\ldots,1,0) \\
(0,0,0,\ldots,0,1)
\end{array}
$$

（重写为“列向量”）将生成密钥 K 的行。实际上，它们是 $N \times N$ 单位矩阵 1_N 的各个列组成的。如果不知道块的大小，则可以使用除了一个“1”以外，其他均为“0”的更长的字符串来推断块的大小。

已知明文攻击更加复杂一些。如果已知块长度为 N，且给定 N 个长度均为 N 的字符串为 $x^1,\cdots,x^N$，且已知加密算法为：

$$y^i = E_K(x^i)$$

目的就是寻找密钥 K（$N \times N$ 矩阵）。将每一个 x^i 和 y^i 重新写成列向量。于是，根据矩阵乘法，我们可以得到：

$$y^i = K \cdot x^i$$

设 X 为 $N \times N$ 矩阵且其列向量为 x^i，且设 Y 为 $N \times N$ 矩阵且其列向量为 y^i，则根据矩阵乘法，我们可以得到：

$$Y = K \cdot X$$

现在，如果只有 X 具有乘法逆矩阵 X^{-1}，我们可以简单地用 X^{-1} 来右乘以该矩阵得到：

$$Y \cdot X^{-1} = K \cdot X \cdot X^{-1} = K$$

这样就可以得到密钥 K。然而，X 不会有乘法逆矩阵的概率至少大约为 0.75。结果就是关系式 $Y = KX$ 不能惟一确定 K。

一种情况是 K 的几种可能必须通过其他的方法单独得到。

另一种情况是我们希望得到不仅仅 N 个已知的明文字符串 x^i，其对应的密文 y^i 也是已知的。在这种情况下，存在更复杂的代数学方法可以运用。

值得注意的一点是选择明文或者已知明文攻击相对来说容易一些（甚至需要更多的数学知识），因为加密和解密是一种矩阵乘法运算。这意味着对于两个块 x 和 x'（重写为列向量）：

$$E_K(x+x') = E_K(x) + E_K(x')$$

对于具有合适大小的矩阵 A, B 和 C，我们可以得到分配率：

$$A(B+C) = AB + AC$$

且对于 $c \in \mathbf{Z}/26$，

$$E_K(cx) = cE_K(x)$$

也就是说，此密码具有**线性特征**。这就是攻击的原理所在。

对希尔密码的唯密文攻击更加困难。只有在我们得到**明密对照**，也就是**可能字**（probable word）时，才能进行该攻击。这就意味着，我们能确信某个特殊的字符会在明文中出现，尽管我们不知道此字符在明文的什么地方出现。实际上，假设攻击者知道明文的一些片断是很合理的，因此认为这是一个可能字攻击而不是一个纯粹的唯密文攻击也是完全合理的。我们将举一个例子，为了简单起见，设分块的大小为 2。

假设使用分块大小为 2 的希尔密码算法 E_k 加密的密文消息为：

QAIXP XDAFG IDYQJ S

并且猜想明文包含的**可能字**为“butter”（例如在“butter and guns”中）。我们将对整个消息进行解密，尽管整个密钥空间具有：

$$(13^2-1)(13^2-13)(2^2-1)(2^2-2) = 157\ 248$$

个可能的密钥（元素属于 $\mathbf{Z}/26$ 的 2×2 矩阵为可逆矩阵）：一个可能字攻击将密钥空间减少到只需要寻找与消息中字符的数目相等的一个集合。

备注 得到密钥空间中的密钥个数，是一些比较普遍的情况中的一个特例。说明什么是正确的不是很困难的事情，但是要解释为什么却是一件困难的事。无论如何，元素属于 $\mathbf{Z}/p^e$ 且具有模 p^e 逆矩阵的 2×2 矩阵的数目为（p 为素数）：

$$p^{4(e-1)} \cdot (p^2-1)(p^2-p)$$

而元素属于 $\mathbf{Z}/p$ 且具有模 p 逆矩阵的 $n \times n$ 矩阵的数目为（p 为素数）：

$$(p^n-1)(p^n-p)(p^n-p^2)(p^n-p^3)\cdots(p^n-p^{n-2})(p^n-p^{n-1})$$

因此，这里存在一些通用的模式，但要给出解释还有些困难。

这里，密钥为 2×2 矩阵 $k = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$，且对明文的每个两字符块 (x_1, x_2) 的加密为：

$$E_k(x_1, x_2) = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

例如，假设分块的分解中具有“bu”和“tt”连在一起进行加密。（如果碰巧“butter”在消息的**偶**数个字符之后出现，则存在此情况。）假设对这两个分块的加密为：

$$E_k(b,u) = (y_1, y_2)$$

$$E_k(t,t)=(y_3,y_4)$$

其中，y_i 为密文字符。这就意味着有：

$$\begin{pmatrix}a & b\\ c & d\end{pmatrix}\cdot\begin{pmatrix}1\\ 20\end{pmatrix}=\begin{pmatrix}y_1\\ y_2\end{pmatrix}\qquad \begin{pmatrix}a & b\\ c & d\end{pmatrix}\cdot\begin{pmatrix}19\\ 19\end{pmatrix}=\begin{pmatrix}y_3\\ y_4\end{pmatrix}$$

或者

$$\begin{pmatrix}a & b\\ c & d\end{pmatrix}\cdot\begin{pmatrix}b & t\\ u & t\end{pmatrix}=\begin{pmatrix}a & b\\ c & d\end{pmatrix}\cdot\begin{pmatrix}1 & 19\\ 20 & 19\end{pmatrix}\begin{pmatrix}y_1 & y_3\\ y_2 & y_4\end{pmatrix}$$

通过求矩阵的逆矩阵，我们就可以解得$\begin{pmatrix}a & b\\ c & d\end{pmatrix}$：

$$\begin{pmatrix}b & t\\ u & t\end{pmatrix}=\begin{pmatrix}1 & 19\\ 20 & 19\end{pmatrix}$$

这里的运算需要模 26，因此通过计算可以得到密钥：

$$\begin{pmatrix}a & b\\ c & d\end{pmatrix}=\begin{pmatrix}y_1 & y_3\\ y_2 & y_4\end{pmatrix}\begin{pmatrix}1 & 19\\ 20 & 19\end{pmatrix}^{-1}$$

当然，为了尝试进行解密，我们实际上所需要的是密钥的**逆矩阵**。因为矩阵比较小，所以可以利用下面公式来求逆矩阵：

$$\begin{pmatrix}a & b\\ c & d\end{pmatrix}^{-1}=\begin{pmatrix}\frac{d}{\det} & \frac{-b}{\det}\\ \frac{-c}{\det} & \frac{a}{\det}\end{pmatrix}$$

因此我们希望：

$$k^{-1}=\begin{pmatrix}1 & 19\\ 20 & 19\end{pmatrix}\begin{pmatrix}y_1 & y_3\\ y_2 & y_4\end{pmatrix}^{-1}$$

类似地，如果“butter” 在消息的奇数个字符之后出现，则对两字符分块“ut”和“te”将被分为一组进行加密。因此，如果我们知道：

$$E_k(u,t)=(y_1,y_2),\qquad E_k(t,e)=(y_3,y_4)$$

则

$$k\cdot\begin{pmatrix}u & t\\ t & e\end{pmatrix}=\begin{pmatrix}y_1 & y_3\\ y_2 & y_4\end{pmatrix}$$

则

$$k=\begin{pmatrix}y_1 & y_3\\ y_2 & y_4\end{pmatrix}\cdot\begin{pmatrix}u & t\\ t & e\end{pmatrix}^{-1}=\begin{pmatrix}y_1 & y_3\\ y_2 & y_4\end{pmatrix}\begin{pmatrix}20 & 19\\ 19 & 4\end{pmatrix}^{-1}$$

密钥的逆矩阵则为（为了进一步的解密所需要）：

$$k^{-1}=\begin{pmatrix}20 & 19\\ 19 & 4\end{pmatrix}\begin{pmatrix}y_1 & y_3\\ y_2 & y_4\end{pmatrix}^{-1}$$

例如，假设明文为“butter???…”。也就是说，假设“butter”恰好是明文的开始。为了进行加密，将明文分成大小为 2 的分块：

bu tt er ?? ??…

然后转换为模 26 的数字，并乘以矩阵$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$。如果这就是明文，那么我们可以得到：

$$E_k(b,u)=k\cdot\begin{pmatrix} b \\ u \end{pmatrix}=\begin{pmatrix} Q \\ A \end{pmatrix},\qquad E_k(t,t)=k\cdot\begin{pmatrix} t \\ t \end{pmatrix}=\begin{pmatrix} I \\ X \end{pmatrix}$$

（密文字符就以“QAIX”开始）或者：

$$k\cdot\begin{pmatrix} b & t \\ u & t \end{pmatrix}=\begin{pmatrix} Q & I \\ A & X \end{pmatrix}$$

因此有

$$k=\begin{pmatrix} Q & I \\ A & X \end{pmatrix}\begin{pmatrix} b & t \\ u & t \end{pmatrix}^{-1}$$

并且通过求逆：

$$k^{-1}=\begin{pmatrix} b & t \\ u & t \end{pmatrix}\begin{pmatrix} Q & I \\ A & X \end{pmatrix}^{-1}$$

$$=\begin{pmatrix} 1 & 19 \\ 20 & 19 \end{pmatrix}\begin{pmatrix} 16 & 8 \\ 0 & 23 \end{pmatrix}^{-1}$$

因为矩阵$\begin{pmatrix} 16 & 8 \\ 0 & 23 \end{pmatrix}$的行列式值可以被 2 整除，所以它不是模 26 可逆的，因此它不可能是一个密钥。

如果明文为“?butter…”，那么可以按照下面的分块方式进行加密：

?b ut te r? ??…

因此，密文的第二块和第三块将是“ut”和“te”：

$$E_k(u,t)=k\cdot\begin{pmatrix} u \\ t \end{pmatrix}=\begin{pmatrix} I \\ X \end{pmatrix},\qquad E_k(t,e)=k\cdot\begin{pmatrix} t \\ e \end{pmatrix}=\begin{pmatrix} P \\ X \end{pmatrix}$$

（因为密文为“QAIXPX…”）或者：

$$k\cdot\begin{pmatrix} u & t \\ t & e \end{pmatrix}=\begin{pmatrix} I & P \\ X & X \end{pmatrix}$$

则

$$k=\begin{pmatrix} I & P \\ X & X \end{pmatrix}\cdot\begin{pmatrix} u & t \\ t & e \end{pmatrix}^{-1}$$

求逆可得：

$$k^{-1}=\begin{pmatrix} u & t \\ t & e \end{pmatrix}\cdot\begin{pmatrix} I & P \\ X & X \end{pmatrix}^{-1}=\begin{pmatrix} 20 & 19 \\ 19 & 4 \end{pmatrix}\cdot\begin{pmatrix} 8 & 15 \\ 23 & 23 \end{pmatrix}^{-1}=\begin{pmatrix} 11 & 14 \\ 9 & 9 \end{pmatrix}$$

但是如果我们使用这个矩阵作为假定的（求逆矩阵）密钥进行解密，我们将得到：

uoutt ehbjv avuwn j

这似乎看来没有什么意义。

如果明文为“??butter…”，则可以按照分块“?? bu tt er ??…”进行加密。

因此密文的第二块和第三块将是“ bu ”和“ tt ”：

$$E_k(u,t)=k\cdot\begin{pmatrix}b\\u\end{pmatrix}=\begin{pmatrix}I\\X\end{pmatrix},\qquad E_k(t,e)=k\cdot\begin{pmatrix}t\\t\end{pmatrix}=\begin{pmatrix}P\\X\end{pmatrix}$$

（因为密文为“ QAIXPX… ”）或者

$$k\cdot\begin{pmatrix}b & t\\u & t\end{pmatrix}=\begin{pmatrix}I & P\\X & X\end{pmatrix}$$

则

$$k=\begin{pmatrix}I & P\\X & X\end{pmatrix}\begin{pmatrix}b & t\\u & t\end{pmatrix}^{-1}$$

求逆可得：

$$k^{-1}=\begin{pmatrix}b & t\\u & t\end{pmatrix}\cdot\begin{pmatrix}I & P\\X & X\end{pmatrix}^{-1}\begin{pmatrix}1 & 19\\20 & 19\end{pmatrix}\cdot\begin{pmatrix}8 & 15\\23 & 23\end{pmatrix}^{-1}=\begin{pmatrix}10 & 9\\11 & 14\end{pmatrix}$$

但是如果我们使用这个矩阵作为假设（求逆矩阵）的密钥进行解密，我们将得到：

eubut tehaj dauus n

这看来也没有什么意义。

如果明文为“ ???butter… ”，则可以按照分块 ??? b ut te r? ??… 进行加密。

因此密文的第三块和第四块为“ut”和“te”：

$$E_k(u,t)=k\cdot\begin{pmatrix}u\\t\end{pmatrix}=\begin{pmatrix}P\\X\end{pmatrix},\qquad E_k(t,e)=k\cdot\begin{pmatrix}t\\e\end{pmatrix}=\begin{pmatrix}D\\A\end{pmatrix}$$

（因为密文为“ QAIXPXDA… ”）或者：

$$k\cdot\begin{pmatrix}u & t\\t & e\end{pmatrix}=\begin{pmatrix}P & D\\X & A\end{pmatrix}$$

则

$$k=\begin{pmatrix}P & D\\X & A\end{pmatrix}\begin{pmatrix}u & t\\t & e\end{pmatrix}^{-1}$$

求逆可得：

$$k^{-1}=\begin{pmatrix}u & t\\t & e\end{pmatrix}\begin{pmatrix}P & D\\X & A\end{pmatrix}^{-1}=\begin{pmatrix}20 & 19\\19 & 4\end{pmatrix}\begin{pmatrix}15 & 3\\23 & 0\end{pmatrix}^{-1}=\begin{pmatrix}15 & 25\\10 & 9\end{pmatrix}$$

如果我们使用这个矩阵作为假设（求逆矩阵）的密钥进行解密，我们将得到：

getbu ttera ndgun s

这个解密即是意义的结果。

习题

8.2.01 假设希尔密码的分组长度为 2，使用已知明文攻击和相应的加密算法：

$$E_K(\text{“guns”})=\text{“YGJC”}$$

密钥K的可能有哪些?

8.2.02 考虑希尔密码，分组长度已知为 2，我们把它当作对双字母的一个代替密码，那么通过对双字母（而不是类似于对维吉尼亚密码那样的单字符）频率统计，对这种密码实施攻击的可能有哪些?

第9章 复 杂 度

在本书的后面部分，我们不想使用过多的复杂度理论，但是从长远的观点看，还是值得把复杂度的概念做一简单介绍。大多数时候，我们实际上只是简单地区分了“快速”和“慢速”算法。从本章中给出的精确含义看来，这在大多数情况下意味着从多项式时间算法中区分非多项式时间算法。在某些情况下，考虑“慢速”算法时，出于某些其他目的还可以细分为“慢速”、“较慢速”和“很慢速”，但是其细节同样对我们来说没有意义。

9.1 大 O 和小 O 符号

对于很多目标，我们不需要准确知道一个数到底有多大，或者一个过程到底有多长。我们只需要知道足够和其他的对象或者程序相比较即可。而且，我们不关心立即比较，而是长期（渐进）比较。这里介绍一些非常基本的概念。

如果 f 和 g 是关于整数或者实数的函数，且存在常数 C 和值 x_0，使得对于所有 $x \geqslant x_0$，

$$|f(x)| \leqslant C \cdot g(x)$$

则我们记作：

$$f(x) = O(g(x)) \text{ 或者简单记为 } f = O(g)$$

即 $f(x)$ 最终以 $g(x)$ 的某个常倍数为界。这个符号既不是说 $g(x)$ 的某个常倍数比 $f(x)$ 要大，也不是说输入值 x 必须要多大才使得这个式子成立。

如果有：

$$\lim_{x \to +\infty} \frac{f(x)}{g(x)} = 0$$

则记作：

$$f(x) = o(g(x)) \text{ 或者简单记为 } f = o(g)$$

如果 $f = O(g)$ 或者 $g = O(f)$，则记作：

$$f \sim g$$

有时也记做 $f = \Theta(g)$，但是后一个符号不如 $f \sim g$ 和“大 O/小 O”符号标准，因此在使用时要注意。

下面给出几个很容易验证但是对于使用“大 O/小 O”符号很方便的规则：

命题

- 如果 $f = O(h)$ 且 $g = O(h)$，则 $f + g = O(h)$；
- 如果 $f = O(h)$ 且 $g = O(k)$，则 $fg = O(hk)$；
- 如果 $f = O(g)$ 且 $g = O(h)$，则 $f = O(h)$；

证明 我们在此只证明第一个命题。假设 $f = O(h)$ 且 $g = O(h)$。设 C_1 和 C_2 为正的常数，且 x_1 和 x_2 为正的实数，则有：

$$|f(x)| < C_1 \cdot h(x) \qquad x > x_1$$

$$|g(x)|< C_2 \cdot h(x) \qquad x > x_2$$

令 x_0 为 x_1 和 x_2 的较大者，且令 $C = C_1 + C_2$，则对于 $x > x_0$，我们可以得到：

$$|f(x)+g(x)| \leqslant |f(x)|+|g(x)|< C_1 \cdot h(x)+C_2 \cdot h(x) = C \cdot h(x)$$

这就证明了第一个结论。 ♣

许多微积分学中的例子都使用了这一概念。其中的一些例子比较基本，但是即使在最坏情况下，还是可以使用 L'Hospital 法则来验证它们。

- $5 = O(1)$，且对于任意常数 c 同样的结论成立：$c = O(1)$
- $x = o(x^2)$，因为 $\lim\limits_{x\to+\infty} x/x^2 = 0$
- $1/x = o(1)$，因为 $\lim\limits_{x\to+\infty} 1/x = 0$
- $x^2+3x+7 = O(x^2)$，因为每一个被加数都是 $O(x^2)$
- $6x^2 = O(x^2)$，因为 $|6x^2 \leqslant 6\cdot|x^2|$
- $\sin x = O(1)$
- 对于任意 $\varepsilon > 0$，有：$\ln x = o(x^{\varepsilon})$
- 对于任意指数 n，当 $x \to +\infty$ 时，有：$x^n = o(e^x)$

习题

9.1.01 证明：$4x^3+3x^2+5x+17 = O(x^3)$。

9.1.02 证明：对于任意整数 $n \geqslant 0$，当 $x \to +\infty$ 时，$x^n = O(e^x)$。

9.1.03 证明：$1+2+3+4+\cdots+(n-1)+n = O(n^2)$。

9.1.04 证明：$1^2+2^2+3^2+4^2+\cdots+(n-1)^2+n^2 = O(n^3)$。

9.2 位操作

描述一个计算要花多长时间，一个较好。且较准确的方法就是给出该计算进行了多少次**位操作**。这种估算忽略了花在读和写过程上的时间，例如存储器访问和寄存器之间的转移，但是这些时间似乎不会严重影响估算结果。一个重要的优点是这些估算忽略了机器的特殊性。而且，甚至在企图对这些算法进行并行计算时还可以再次利用这种估算。

在本节中，我们关注的重点是，对于一些基本算术运算（例如加法和整数乘法）的**复杂度**，如何找到一种很好的近似。

“行为”的最小单元将被比较模糊地定义为**位操作**。通常，一个**位**（“二进制数字”）是信息的最小单元，且只能为 0 或者 1。一个位操作有两个输入，每一个输入要么为 0，要么为 1。有两种输出，每一个输出要么为 0，要么为 1。我们不关心这些 0 和 1 存储在什么地方，或者它们是如何移动的。

在这个简单的模型中，为了按照 0 和 1 来描述算术自动，最好是将整数写作二进制形式，尽管我们并不打算实际上以二进制形式进行任何计算。尤其没有必要过多考虑整数的十进制表示与二进制表示之间的转化算法。这一转换算法在任何实际的算法中很少起到主要作用。但是，对这个变换进行一些思考是值得的,这样我们就应该仔细考虑关于表示整数的符号体系。

通常的十进制表示将整数表示为 10 的方幂的倍数（从 0 到 9）的和，例如：

$$1375 = 1375_{10} = 1\cdot 10^3 + 3\cdot 10^2 + 7\cdot 10^1 + 5\cdot 10^0$$

如果为了明确，或者是为了强调，通常需要写出下标。同一个数（十进制 1375）的二进制表示为：

$$10101011111 = 10101011111_2 = 2^{10} + 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$$

使用 2 的幂而不是 10 的幂，且系数也在相应的范围内，这里为 0 或者 1。因为这是最小的范围，所以如果愿意，我们可以完全省略系数，而且可以省略系数为 0 的 2 的方幂。现在对于使用其他的非 2 或者非 10 的“基”进行类似的计算已经很清楚了。八进制和十六进制的应用也已经比较普遍，我们的计算机系统中的时间对于分和秒都是 60 进制，这是主要受到古代巴比伦人的影响（以及其他!）。

如何获得一个整数的基为 b 的展开式，这个**算法**也很简单。首先，我们给出一些术语：给定一个整数 n 和一个正整数 d，n/d 的**整数部分**为小于或者等于 n/d 的最大整数。给定一个正整数 n，以 b 为基可以将 n 表示为：

$$n = a_0 + a_1\cdot b + a_2\cdot b^2 + a_3\cdot b^3 + \cdots$$

（每一个“数字” a_i 均在范围 $0 \leqslant a_i < b$ 内），执行下列步骤：

$a_0 = n\%b$ 且用 n/b 的整数部分来替换 n

$a_1 = n\%b$ 且用 n/b 的整数部分来替换 n

$a_2 = n\%b$ 且用 n/b 的整数部分来替换 n

…

继续执行直到 n/b 的整数部分变为 0 为止。

备注 在前面的过程中，用一个新值“替换” n 的方法非常方便。与之对比，如果我们坚持对 n 的每一个“新”值进行命名，那么除了多出几个名称之外，没有任何用处。

加法和乘法在二进制中的通用规则与十进制中是很类似的，只不过在二进制中“单数字”规则在数量上很少且很容易记忆：

$0+0=0,\quad 1+0=0+1=1,\quad 1+1=10$

$0\cdot 0=0,\quad 1\cdot 0=0\cdot 1=0,\quad 1\cdot 1=1$

多位二进制整数的*加法*计算与对十进制整数的计算一样，数字方式是从右到左，适当的时候进位。同样，二进制中的多数字乘法的通用方法也和十进制中一样。

两个单数位二进制整数的加法操作是位操作的一个例子：第一个输出位是两个输入位的和，另一个输出位则是两个输入位的和的进位。同样，两个单数字二进制整数的*乘法*操作也是一个位操作：在此情况下，我们只使用其中一个输出位，它是两个输入位的乘积结果。（在二进制中单数字乘法的计算不存在任何进位。）

如果在二进制中需要用 n 个数字来表示一个整数 x，则我们就说它是一个 ***n* 位整数**。也就是说，n 位整数 x 就是那些在 $2^n - 1 \leqslant x < 2^{n+1}$ 之间的数。因为 $\log_{10} 2 \approx 0.30103 \approx 3/10$，所以二进制数字的个数是十进制数字的个数的 10/3 倍。

命题

- 比较两个 n 位整数（决定哪一个更大，或者它们是否相等）需要执行 $O(n)$ 次位操作。
- 两个 n 位整数的加法需要执行 $O(n)$ 次位操作。
- 一个 m 位和一个 n 位整数的乘法需要执行 $O(mn)$ 次位操作。
- 对 n 位整数进行模一个 m 位的整数的带余除法需要执行 $O(mn)$ 次位操作，其中 $m \leqslant n$。

备注 根据简单的小学算术即可以证明！

具有小于或者等于 n 位的数字的个数最多有 2^n 个。因此，如果我们不是根据二进制数字的个数来描述数字，而是根据它们的大小，则我们可以将上述命题改写为：

命题

- 比较两个小于 n 的整数的大小需要执行 $O(\log_2 n)$ 次位操作。
- 两个小于 n 的整数的加法需要执行 $O(\log_2 n)$ 次位操作。
- 两个小于 n 的整数的乘法需要执行 $O(\log_2{}^2 n)$ 次位操作。
- 对一个小于 n 的整数进行模一个小于 m 的整数的带余除法需要执行 $O(\log_2 m \log_2 n)$ 次位操作，其中 $m \leqslant n$。

通常我们有：

$$\log_{10} 2 \cdot \log_2 n = \log_{10} n$$

且 $\log_{10} n$ 是 n 的十进制数的位数。因此如果一些计算对于输入整数 n，需要执行 $O(\log_2{}^l n)$ 次位操作，则它需要执行 $(\log_{10} 2)^l \times O(\log_{10}{}^l n) = O(\log_{10}{}^l n)$ 次位操作。这里的部分观点就是，*在这种估算中不用考虑使用哪一种算法*。因为从基 2 转化为基 10（或者基为 e）不会改变大 O 的估算结果。

备注 特别对于大整数，存在几种乘法的算法，其运算速度比在小学里学到的算法要快得多。但是，只有所选整数比小学孩子们所能使用的数字大得多时，才会产生实际的加速效果。

备注 对于人们的使用而言，使用二进制形式表示显得非常不经济：乘法和加法表很小，但是必须执行非常多的操作（毫无疑问简单得多）：一个 n 位数的十进制数将有超过 $3n$ 个的二进制数字，因此一个二进制乘法（使用最佳算法）将包含 9 倍于十进制乘法的操作，虽然每一个单独的操作都比较简单。*另外存储空间问题也不可低估*。

尽管人们执行这些基本算法在文字上的最优化似乎并不重要，但是在计算问题上，时间/存储空间的平衡是很重要的。通常我们容易为时间问题担心，而不是空间问题，但实际上后者也应考虑。

习题

9.2.01 估算对前 2^{10} 个整数进行累加（最优估计）所必须的位操作数量。

9.2.02 估算对前 2^{10} 个整数进行累加并模 101 约简（最优估计）所必须的位操作数量。

9.2.03 估算对 100 !进行因式分解（最优估计）所必须的位操作数量。

9.2.04 估算计算 100 ! mod103（最优估计）所必须的位操作数量。

9.2.05 估算计算 2^{100}（重复计算 $2^{n+1} = 2^n \cdot 2$）所必须的位操作数量。

9.2.06 估算计算 $2^{100}\%117$（重复计算 $2^{n+1} = 2^n \cdot 2$）所必须的位操作数量。

9.2.07 估算对一个具有 m 位和一个具有 n 位的数进行欧几里得运算所必须的位操作数量。

9.2.08 估算对一个 n 位的数字进行直接的素性测试所必须的位操作数量。

9.3 概率算法

对很多人来说，一个算法中存在随机元素的情况会令他们感到惊异。对于一些更加重要

的问题来说，使用概率算法比使用确定算法要好，这使得人们更加惊异。这里有一些术语：

蒙特卡罗算法总是给出答案（在多项式时间以内），但是其中的结论是以一定的概率而成立的。更确切地说，“yes”回答只有在一定概率下是正确的，但是“no”回答始终是正确的，则称之为**有偏**（与之相反我们可以得到**无偏**蒙特卡罗算法）。因此，一个真正的蒙特卡罗算法要么是有偏的，要么是无偏的，因此“半数”的回答是确定的。

拉斯维加斯算法只运行在期望的多项式时间内，因此可能不会给出任何答案，但是如果它针对一个特定输入给出一个回答，则该答案一定是正确的。

一个不太标准的用法是，一个**大西洋城**算法至少在 3/4 时间内给出正确答案，且运行于多项式时间内。（数字 3/4 可以被任意一个大于 1/2 的概率代替。）

大西洋城算法是蒙特卡罗算法的双边形式。

9.4 复杂度

一个算法的复杂度最简单的度量是，它作为输入值大小的一个函数要运行多长时间。因为我们主要关注在大输入情况下会发生什么情况，我们将使用大 O/小 O 概念，从而不用担心微小的细节部分。

对于某个实数 r，且输入的长度为 n，如果一个算法在 $O(n^r)$ 时间内结束，则称其在**多项式时间**内完成。这种算法通常被认为是好的。注意在此时间估算中指定的常数必须与 n 独立。这意味着即使在最差的情况下（例如在狡猾的或者肮脏的输入下），估算也必须成立。通常完全有可能使用更少的时间，但是必须考虑最差的情况。

如果一个问题具有一个在多项式时间内运行就可以解决的算法，则称该问题属于**类** *P*。

如果对一个问题答案做出猜测，其正确性可以在多项式时间内证明或者排除，则该问题属于**类** *NP*。这个类必定包含类 *P*。

人们普遍认为类 *P* 严格小于类 *NP*，但这是复杂度理论中的最主要的一个公开问题。在 *P* 类中的问题被认为是**简单的**，属于 *NP* 但不属于 *P* 的问题称为**困难**问题。具有讽刺意义的是，因为我们还不能证明类 *P* 严格小于类 *NP*，所以到现在我们都还不能证明，存在这个意义上的任何困难问题。

当然，在任何情况下当我们使用渐进的大 O/小 O 符号时，我们都忽略了常数。因此如果对于输入长度为 n 的情况，最差的运行时间小于或者等于：

$$10^{1000} \times n^2$$

则我们就有了多项式运行时间，但是前面的常数很大，以至于该操作永远不能结束。这种可能性在这里实际上没有考虑！

如果有一个多项式时间算法能够根据对问题 B（关于有相关的输入）的回答得出问题 A（对于给定的输入）的回答，则我们说问题 A 可简化为问题 B（在多项式时间内）。据此，一个对问题 B 的未知或者不确定的处理称为问题 B 的 **oracle**。根据直觉，我们知道 A 问题不会比 B 问题复杂。令人惊奇的是，已经证明存在问题 C 属于 *NP*，使得属于 *NP* 的所有问题都可以在多项式时间内简化为 C。这些问题称为 ***NP* 困难问题**或者 ***NP* 完全问题**。（与我们使用的不同，更加精确的区分可以使得这两个概念代表不同的事物。）

其他类型的问题是那些可使用**随机化**在多项式时间内回答的问题。类 *ZPP* 包括那些使用拉斯维加斯算法在多项式运行时间内是可回答的问题。类 *RP* 包括那些使用蒙特卡罗算法在

多项式运行时间内是可回答的问题。那些使用大西洋城算法且在多项式运行时间内是可回答的问题包含在类 *BPP* 中。符号 *P* 和 *NP* 是完全标准的用法，而 *ZPP*、*RP* 和 *BPP* 都不能算是标准的用法。

如果一个算法不能（最差情况下）在多项式时间内运行完毕，则（缺省状况）我们称之为运行在**指数时间**内。注意这也是根据输入的大小来度量的。

因此，例如回顾在使用直接的试除法来证明数 N 的素性时，只需要使用 $\sqrt{N}$ 步就可以证明 N 是素数（如果它是素数的话）。N 的大小 n 作为输入，即 $n=\log_2 N$，且有：

$$\sqrt{N}=2^{\log_2 N/2}=2^{n/2}$$

这比 n 的任何多项式都要快得多。与此对比，甚至对整数的加法和乘法常用的直接算法也运行于多项式时间内。我们迫切希望得到运行时间的估算方法，独立于算法运行的机器种类，以及编写程序的语言。当然，这必须意味着只要具备与合理期望一样的独立性，因为在执行大多数算法时，一个运行速度较快的机器比一个运行速度较慢的机器要快。使得这个假设更加精确的一个方法是要求使用“足够好的”计算机（非并行机）。

假设 算法的运行时间仅仅取决于其必需的**位操作**的数量以及**机器的速度**，用公式表示如下：

$$\text{运行时间}=\frac{\text{位操作数量}}{\text{机器速度}}$$

备注 该假设的大概意思是机器可能更快或者更慢，但是不断提高机器的执行速度并不是一种明智的做法，E. Post、Kurt Gödel、Alan Turing 和 Alonzo Church 的工作表明，如果我们只考虑并非根本原因的机器以及运算序列中指令的执行，那么这个假设就是正确的。另一方面，在**无穷多个并行**计算机上的做**无穷多个并行**算法，例如**量子计算机**（如果存在的话），就不属于这种类型，因此会做得“更好”。

备注 关于是否存在比仅考虑速度更好的方法，这个问题很有意思。算法质量也起到很大作用：与一台慢速计算机运行一个好的算法相比较，一台快速机器运行一个慢速算法可能会更差。机器的功能表现出通常对于人来说是“聪明的”特征，这是很好笑的事情。

9.5 子指数算法

通常只有一个算法是多项式时间算法时，它才被认为是一个好的算法，也就是说，它的运行时间是输入的大小的多项式。但是，根据这一点来说，将整数分解为素数的乘积、计算离散对数和以及其他一些重要的计算的最好的算法都不是多项式时间的，而且不会比指数时间快多少。为了实用的目的，这可能产生决定性的区别。我们可以对那些比多项式时间慢但比指数时间快的算法速度之间的差异进行量化。

算法类 $L(a,b)$ 是这样一类算法，它对于输入大小 n 的运行时间为：

$$O(e^{(b+o(1))\cdot n^a(\ln n)^{1-a}})$$

其中，$b\geqslant 0$，$0\leqslant a\leqslant 1$。这里的 $o(1)$ 代表一些函数，当 $n\to\infty$ 时，该函数趋近于 0。

对于所有 $0<a<1$，$0<b$ 的空间，并集 $\bigcup_{0<a<1,0<b} L(a,b)$ 为**子指数算法**的集合。

考虑一个极端情况，对于任何 $b\geqslant 0$，$L(0,b)$ 是运行时间为 $O(n^b)$ 的多项式时间算法的集合。另一个极端情况是，每一个集合 $L(1,b)$ 包含运行时间为 $O(e^{bn})$ 的指数时间算法。

目前已知的因式分解的最好算法和在 $\mathbf{Z}/p^e$ 中计算离散对数的最好算法都被推测属于 $L(\frac{1}{3},1.923)$ 类。

分解因式见[Coppersmith1993]，$\mathbf{Z}/p^e$ 离散对数的计算见[Adleman1994]。注意这些运行时间估算只是一种推测，因为它们依赖于关于一些特殊类型整数分布的似是而非但是未经证明的假设。而且，比较特殊的是这两个运行时间的估算是相同的。特别需要注意的是，这个离散对数运行时间的估算只有在最简单的有限域 $\mathbf{Z}/p$ 中才成立。对于一般的有限域，最好的估算似乎为 $L(\frac{1}{2},b)$。而且，对于更多的抽象离散对数问题，例如在椭圆曲线上，不知道是否存在这样低的运行时间。

9.6 柯尔莫哥洛夫复杂度

最近，Solomonoff、Chaitin 和 Kolmogorov（柯尔莫哥洛夫）（独立地）提出了一套复杂度理论，强调程序长度（针对字符数）而不是程序运行时间。这个观点也针对随机数和更广义的概率提出了一个不同的观点。根据这个观点，他们对信息论以及关于压缩的问题也进行了阐述。

这种思想即为，对一个比较特殊的字符串，可用比将其全部列举出来更加简单的方法来描述，例如：

10101010101010101010101010101010101010101010101010101010

可以描述为“26 个重复的 10”。但是，下面的字符串由于没有明显更加简单的描述方法，因此还是采用列举的方法：

10111011101000001010000011011010011111111011000000110

从概率论的观点来看，有时我们假设每（例如）56 个 0 和 1 组成的字符串都是等可能随机选择的。对于上述两个字符串，第二个似乎是“随机”的，而且如果我们被告知它确实是随机选择的，我们也绝对不会怀疑。但是，上述第一个字符串“太结构化”，因此我们将怀疑它“可能曾经是”随机选择的。传统的目标是，对于 2^{56} 个 0 和 1 组成的字符串中，假设证明这个字符串不是随机选择的，那么字符串由 26 个“10”组成（即为 2^{-56}）的概率是非常小的。但是，根据这个观点，第二个字符串也是同样不太可能，但是我们并不反对它！

我们可以将关于“长度为 n 的随机字符串”的概念修改为包括这样一种思想，即对一个真正随机对象的描述不应该比其本身更简单（或短）。这就使得我们拒绝这个有 26 个“10”组成的字符串更加合法，因为它的描述太短，而接受另一个字符串为“随机的”，因为无法用一个比较简单的描述来表达。

柯尔莫哥洛夫复杂度的第一个非正式定义为：完全描述对象所需要的最短长度。这里“对象本身”可以当作是它本身的一个描述。当然，对对象的冗长描述是没有任何意义的。通常的问题为是否存在一种比其本身更短的重要描述。

可以证明（使事物更加精确之后！）：给定长度为 n 的“大多数”字符串的描述，不会比 n 更短，因此，对它们的最短的描述往往是将其列举出来。

当然，对这个问题严格处理的一个必要准备就是证明程序长度仅以一种肤浅的方式依赖于特定的机器和编程语言。

在使用这个观点时，为了说明需要十分小心（以及精确的形式化），我们按如下方式定义一个正整数 n：

$$n=\text{不能用少于 60 个字符来描述的第一个整数}$$

矛盾的是，我们明显只需要少于 60 个字符来描述它。解决矛盾的方法之一是声明每一个整数都只用少于 60 个字符来表示。但这是一个站不住脚的声明：因为只有有限多个字符串的字符数量少于 60（使用由字母 a 到 z，数字 0 到 9，空格符和标点符号组成的有限集合）。实际的缺点可能在于“描述”一词的意义必须更加准确。

一个更加完整的关于这个主题的论述可以参考 M. Li 和 P.M.B. Vitanyi 的论文：Introduction to Kolmogorov Complexity and its Application, Springer, 1997(second edition), ISBN 0387948686。

9.7 线性复杂度

一个由 0 和 1 组成的有限序列的柯尔莫哥洛夫复杂度的判定通常不可能在多项式时间内解决。一个可能的解释是，求解对象的柯尔莫哥洛夫复杂度是一个太令人激动以及太普通的问题。从另一个极端来看，不是通过任意过程生成由 0 和 1 组成的（有限但很长的）序列，人们可能要求通过**线性反馈移位寄存器**（LFSRs）生成由 0 和 1 组成的序列。我们将在后面详细讨论这些问题，但是为了目前的目的，我们在这里可以足够详细地描述这些事情。

一个**长度**为 N 的 LFSR 为一个由 0 和 1 组成的序列：$c=(c_0,\cdots,c_{N-1})$。我们还需要一个初始状态：$s=(s_0,s_1,s_2,s_3,s_{N-1})$，则我们进行递归定义，对于 $n+1\geq N$，有：

$$s_{n+1}=c_0s_n+c_1s_{n-1}+c_2s_{n-2}+\cdots+c_{N-1}s_{n-3}\%2$$

这个机制比在柯尔莫哥洛夫复杂度中所允许的完全任意的过程要简单得多。

一个**有限序列** $x=(x_0,x_1,\cdots,x_t)$ 的线性复杂度为是使得对于选定的初始状态 $s=(s_0,s_1,s_2,s_3,s_{N-1})$，生成序列 x 所需要的线性反馈移位寄存器的最小长度为 N。也就是说，与其要求根据任何种类来对序列进行最小描述，还不如严格定义产生机制的类别，例如 LFSRs。因此，一个序列的线性复杂度完全有可能很高，尽管从柯尔莫哥洛夫复杂度的观点来看可能很低。一个序列的线性复杂度很低，则其柯尔莫哥洛夫复杂度也一定很低。

与柯尔莫哥洛夫复杂度显著不同，有限序列的线性复杂度的判定有一种相对简单的算法。也就是，判定生成该序列所需要的最小 LFSRs。根据 Massey 和 Berlekamp 的工作，在仅仅抽取序列中 $2N$ 个连续的项之后，这个算法将决定长度为 N 的 LFSR 的系数。

后面我们将看到，长度为 N 的 LFSRs 在重复自己之前可以生成长度为 2^N-1 的序列，此特点（错误地）使得人们将其用于密码学目的。关键在于，Massey 和 Berlekamp 算法表明，尽管在重复之前需要很长时间，但是对于密码学的用途来说，其线性复杂度还是太低。

9.8 最差情况与期望值

在讨论算法的运行时间时，必须记住的一个特性是在*最差情况*与*期望*运行时间之间的对比。但这与密码的*构造*有关，因为加密和解密都不能太困难。这个区别与*密码分析学*有关（可能是敌对的），因为不管在什么情况下，人们手中掌握的东西不可能是“平均”值，也不可能是最差情况。

进一步说，就算可以证明一个密码的平均安全性，也不能防止其某一个特例被成功攻破。

这一点对于非对称（公钥）密码算法和对称密码算法都是正确的。例如，存在**弱密钥**问题：即使是一个相当好的密码，其使用中也要避免选择一些特殊的密钥，这种情况很普遍，因为如果使用这些密钥，就会大大降低该密码算法的强度。

当更多地谈及传统的复杂度理论时也必须记住这个区别，因为只对最差情况进行处理是一个标准。因此，在考虑很多定理或者传统复杂度理论的其他结论对于密码问题的适用性时，往往会给出一个不恰当的印象。

第10章 公钥密码算法

本章介绍了公开密钥密码（非对称）算法。本书的其余部分将对其进行详细介绍和解释，研究方法是对这种密码进行的攻击。

另一术语“非对称”揭示了这种密码算法的一些特点。首先，相比较而言，所有的传统密码以及特定的现代密码算法（例如 DES 和 AES）都是一种对称算法，即解密密钥相当于（通常完全等于）加密的密钥。与此相反，加密和解密的密钥在非对称密码算法中是不相同的（通过可行的计算）。也就是说，如果加密和解密密钥其中之一保密，而另一个公开，允许多个不同的人进行加密，但是只有一个人能够进行解密。了解公开密钥密码算法的非对称性对于理解一些新出现的问题是一个关键。

Diffie-Hellman 密钥交换、RSA 密码、ElGamal 密码以及 knapsack 密码都是著名的、标准的和相对基本的密码。Adi Shamir 对 knapsack 密码的攻击，以及随后对这种攻击的防御，在这里进行解释都较为复杂，而且远离我们讨论的主题。我们介绍了一个新的且相当有发展前途的密码，即 NTRU 密码，其机制相对不是那么初等，而且其安全性依赖于更加成熟的东西。关于 NTRU 密码的有关章节相对前面内容来说比较复杂，但还是基于我们已经介绍过的内容，或者后续章节将要介绍的内容。最新也是最复杂的讨论，即关于算术密码以及针对它进行 Hughes-Tannenbaum 攻击的内容，对于初次阅读这部分内容的读者来说，可以完全跳过。

大约 1975 年之前，惟一存在的一类密码就是**对称密钥密码**，意味着知道了加密密钥就很容易得到解密密钥，相反亦然。这种密码通常也称为**秘密密钥密码**，因为所有的密钥都必须保密。

与此相反，**公开密钥（非对称）**密码体制的加密密钥与解密密钥没有本质的关系。考察所有的古典密码，确实不能给出任何公钥密码是可能存在的暗示。

最初在 Ralph Merkle 进行了大量（未被欣赏的）工作之后，公钥密码的大致思想首先在[Diffie, Hellman 1976]中公开被提出来。（实际上英国秘密服务机构的一些人在 20 世纪 60 年代中期就提出了类似的思想，这个秘密直到 20 世纪 90 年代末期才被披露出来。）一个基于**knapsack 问题**的公钥系统在[Merkle，Hellman 1978]中被提出。后一个系统已经被“攻破”，尽管它现在得到了“修正”，但是在 knapsack 问题上失掉的信心似乎很难改变。最流行的一个公钥密码系统是 RSA，以三个作者的名字命名[Rivest，Shamir，Adleman 1978]。RSA 的安全性基于整数素因子分解的困难。ElGamal 系统[ElGamal 1985]是一个出现相对迟一些的密码，首次出现于 1985 年。

公钥密码系统在应用上的可能在以前是不可想像的，我们将在后面讲到这些**协议**。

使用公钥密码的一个简单的例子是在**通信网络**中。有 N 个人互相之间使用非对称密码（例如 RSA）进行通信需要 N 个三元组 (e,d,n)：每一个人公布他自己的公钥。因此，为了与其他人安全地通信，每一个人都只需要使用相应的公钥进行加密即可。也就是说，整个通信网对于每一个人来说只需要一批信息 (e,d,n)。与之相反，一个对称密码算法将在每两个人

之间需要一个密钥对，对于 N 个人来说总共需要 $N(N-1)/2$ 个密钥对。因此，在维持通信网络时使用非对称算法可以极大地降低密钥的数量。对于包括 N 个人的通信网络，使用对称密码需要 $N(N-1)/2$ 个密钥对，其中每一对人之间都有一个加密/解密密钥对是不好的。首先，网络中的每一个人都必须记住 $(N-1)$ 个密钥；其次，总共有 $N(N-1)/2$ 个密钥对需要生成和分发。

因为非对称密码的加密和解密算法要比对称密码的加密和解密算法慢得多，在实际应用中非对称密码算法通常被用来安全地交换对称密码的**会话密钥**，而这个对称密码被实际用来加密通信。也就是说，被非对称密码加密的惟一的明文就是对称加密密码的密钥，因此具有更快运行速度的对称加密密码被用来对实际的信息进行加密。

使用会话密钥的技巧目前在密码系统的使用中十分通用。也就是说，公钥密码系统被用来为（更快速的）私钥密码系统建立共享密钥（会话密钥），这样将产生大量的通信量。消息被发出以后，会话密钥就被废弃不用了。因此，对秘密信息进行加密时，公钥密码的优点可以实现，同时也可以受益于对称密码运行速度快的特点。进一步，在应用中对于新的“外来”协议，没有任何其他方法可替代公钥密码。

10.1 陷门

每一个非对称（公开密钥）密码算法都依赖于实际某种处理过程的不可逆性，通常这指的是**陷门**。目前，所有理论上认为安全的非对称密码都使用数论，尽管在原理上还有很多其他的可能性。

RSA 密码利用的原理是，计算两个大的素数 p 和 q（可能 $\approx 10^{80}$ 或者更大）的乘积 $n=pq$ 不难，但是将一个大的整数 $n \approx 10^{160}$ 分解为其素因数的乘积几乎不可能。

ElGamal 密码利用的原理是，对于模一个大的数 m 进行求幂不难，但是计算离散对数非常困难。给定 x,e,p（p 为素数），所有数都大约为 $\approx 10^{140}$，计算 $y=x^e \% p$ 并不是很难。但是从另一个方向，给定 y,x,p 计算指数 e（y 是以 x 为基模 p 的离散对数）似乎很困难。

注意在最后两段中的限制：我们说这个任务似乎是很困难的。到目前还没有人证明将整数分解为素数的计算在本质上是非常困难的。从另一个方面来说，存在大量的实际证据表明这是一个困难问题：人们思考这个问题已经有几百年了，由于与密码有关，因此这个问题最近越来越受到关注。离散对数问题也是一样的情况。

最早的公钥密码为 Hellman-Merkle，具有相反的问题：基于一个可能的困难问题，即所谓的 knapsack 问题，对其进行了必要的修改，使得解码有可能很大程度上改变此问题，从而使得此问题不再是一个难题！

在 1978 年，McEliece 提出了一种基于代数编码理论的密码。这种使用 Goppa 编码的算法似乎是一种一般线性编码。对于有线性编码的解码被证明是可能困难（“完全 NP”）的问题，而 Goppa 编码的解码问题是很“容易”的。这种密码算法似乎还没有被攻破，但是没有 RSA 和 ElGamal 那么流行。将一个更加简单的问题“隐藏”到一个完全 NP 问题中去的思想与 Hellman-Merkle 中的技巧类似，这种方法曾经使一些人紧张且对 McEliece 密码产生了怀疑。

因此，最后尽管这些问题没有被证明是困难问题，因式分解和求解离散对数问题的明显实际的困难性（在每一次详细审查之后）使得 RSA 和 ElGamal 至今仍然是最流行的公钥密

码。最近发表的**椭圆曲线密码**是离散对数密码的一种抽象变化。对其进行详细的描述需要大量的深入准备工作。

10.2 RSA 密码

RSA 密码的思想取决于[Rivest,Shamir,Adleman 1978]。其关键之处在于将大数分解成为素数乘积是很困难的。可能很令人奇怪，仅仅对大整数测试其素性而不找出其因子要简单得多。

这里的**困难任务**就是将大的整数分解成为素数因子的乘积。基本任务相对简单一些，即：

- 对于 $n>10^{160}$ 且大的指数 e，求幂：$x^e\%n$ 模 n；
- 找到许多大的素数 $p>10^{80}$。

我们可以看到，这些任务的明显困难性是 RSA 密码算法安全性的基础。

将大的整数分解成为素数因数的乘积很困难，这在直觉上是很明显的，尽管还没有证明本身的困难程度。比较而言，我们可以对大整数的素性进行测试而不需找到它们的因子。关于有效地评估大整数 x 的大指数 x^e 模大整数 n 约简是更基本的事情。记住，当计算速度提高的时候，相应地所选择的整数 $n>10^{160}$ 和 $p>10^{80}$ 也应相应增加，即使算法上并没有任何的改进。

加密和解密的描述：有两个密钥 e 和 d，非秘密的附加信息由一个大的整数 n 组成（n 的特性，e 和 d 之间以及与 n 的关系将在下面描述）。明文 x 首先被编码为正整数，我们仍然称之为 x，因为现在的原因，我们要求 $x<\mathrm{n}$。因此加密步骤为：

$$E_{n,e}(x)=x^e\%n$$

其中 $z\%n$ 表示 z 模 n 的**余数**。这里就生成了密文 $y=x^e\%n$，它也是一个正整数，范围为 $0<y<n$。**解密**步骤很简单，为：

$$D_{n,d}(y)=y^d\%n$$

例子 为了简单起见，我们人为地使用较小的数。Alice 选择两个素数 $p=71$，$q=59$，则 $n=4189$，其中 p 和 q 都为模 4 同余 3 的。Alice 决定使用 $e=3$ 作为公钥。她将 $n=4189$ 和 $e=3$ 公开，并对 p 和 q 保密。为了得到解密指数，她需要找到 e=3 模 $(p-1)(q-1)$ 的乘法逆元，于是她可以通过使用欧几里得算法得到：

$$d=e^{-1}\bmod 4189=2707$$

（或者在数比较小的情况下强行计算得出）。当 Bob 希望给 Alice 发送一条秘密消息的时候，例如，$x=1234$，他计算密文：

$$y=x^e\%n=1234^3\%4060=229\bmod 4189$$

他将该数发送给 Alice。当 Alice 接收到这个消息后，通过下面计算进行解密：

$$x=y^d=229^{2707}\%4189=1234$$

因此，她可以得到明文。

当然，在解密步骤中，两个密钥 e 和 d 对于所有 x（至少满足范围 $0<x<n$）必须具备以下特性：

$$(x^e)^d\equiv x\bmod n$$

欧拉定理（下面将要讨论）声称通常如果 $gcd(x,n)=1$，则有：

$$x^{\varphi(n)}\equiv 1\bmod n$$

其中 $\varphi(n)$ 是对 n 求欧拉函数得到的值，$\varphi(n)$ 定义为在 $0<l\leqslant n$ 内满足 $gcd(l,n)=1$ 的整数 l 的数目。因此，e 和 d 之间的关系是以 $\varphi(n)$ 为**模的乘法互逆关系**，即：

$$d\cdot e\equiv 1 \bmod \varphi(n)$$

对于形为 $n=p\cdot q$ 的整数 n，其中 $p\neq q$，这个函数的值有一个简单的表示方法（我们在后面会了解）：

$$\varphi(p\cdot q)=(p-1)(q-1)$$

这解释了上述表达式 $(p-1)(q-1)$ 出现的原因。在这种情况下，我们可以验证加解密确实是工作在条件 $gcd(x,n)=1$ 下的：

$$D_{n,d}(E_{n,e}(x))=(x^e\%n)^d\%n=(x^e)^d\%n$$

因为到现在我们知道，只要我们希望得到，无论什么时候都可以做模 n 的约简，在算术计算上，最终得到的结果也是模 n 约简的。根据指数的特性，

$$(x^e)^d\%n=x^{e\cdot d}\%n$$

因为 $ed\equiv 1 \bmod \varphi(n)$，存在一个整数 l 满足：

$$ed=1+l\varphi(n)$$

于是根据欧拉定理，有：

$$x^{ed}=x^{1+l\varphi(n)}=x^1\cdot(x^{\varphi(n)})^l\equiv x\cdot 1^l\equiv x \bmod n$$

注意我们必须假设明文 x 与 n 互素。因为 n 是两个大素数 p 和 q 的乘积，与 n **互素**即意味着不能被 p 或 q 整除。在范围 $0\leqslant x\leqslant n$ 内的一个“随机”整数 x 可以被 p 和 q 整除的概率为：

$$\frac{1}{p}+\frac{1}{q}-\frac{1}{pq}$$

这是一个非常小的数，因此我们忽略这个概率。

加密的指数 e（和解密的指数 d）必定与 $\varphi(n)=(p-1)(q-1)$ 互素，因此它就一定具有一个模 $\varphi(n)$ 的乘法逆元，这个逆元即为解密指数 d。

通常所见到的应用RSA密码体制的一个事件链如下所述：Alice选择两个大的素数 p 和 q（其中 $p\neq q$），p 和 q 都是模 4 同余 3 的，并计算 $n=pq$。素数 p 和 q 必须保密。她又选择一个加密和解密指数 e 和 d 满足 $e\cdot d\equiv 1 \bmod \varphi(n)$。她在她的 Web 网页上公布加密指数 e 以及模数 n，她的解密指数 d 也保密。如果任何人想要给她发送只有 Alice 才能阅读的加密电子邮件，就可以对明文 x 进行加密：

$$E_{n,e}(x)=x^e\%n$$

Alice 是惟一一个知道解密指数 d 的人，因此她是惟一一个可以还原出明文的人，还原步骤如下：

$$x=D_{n,d}(E_{n,e}(x))$$

因为在这种情况下，她可以使得加密密钥公开，通常加密密钥 e 被称为**公钥**，解密密钥 d 被称为**私钥**。

备注 具有形式 $n=p\cdot q$ 的整数 n 被称为 **Blum 整数**，其中 p 和 q 是不同的素数，且都是模 4 同余 3，n 有时也被称为 **RSA 模数**。

备注 很明显，Alice 必须能够计算整数模 4189 的比较大的方幂（她对于加密指数 3 的选择意味着加密者无须担心这个问题）。我们将在后面看到，有一个很好的算法来执行这个求幂运算。

RSA 的安全性的基本方面。RSA 的安全性或多或少依赖于将整数分解为素因子乘积的难度，而这似乎确实是一个十分困难的问题。但是，更加精确地说，RSA 的安全性依赖于一些更加特殊的问题，即对特殊形式的数 $n=p\cdot q$（p 和 q 都是素数）进行因式分解的难度。可以想像得到，可以通过不适用于一般问题的特殊方法来解决一些更加特殊的问题。但是到现在，问题的特殊性似乎并没有产生任何特别好的特殊因式分解攻击。

因式分解的难度使得 RSA 是安全的，原因在于对于两个大素数 p 和 q（该两素数保密）的乘积 n，在只知道 n 的情况下计算 $\varphi(n)$ 似乎很难。当然，一旦知道素数分解 $n=p\cdot q$，则通过下面的标准公式很容易计算得到 $\varphi(n)$：

$$\varphi(n)=\varphi(p\cdot q)=(p-1)\cdot(q-1)$$

如果攻击者知道 $\varphi(n)$，则根据**欧几里得算法**由加密指数 e 计算解密指数 d 相对容易一些，因为解密指数刚好是 e 模 n 的乘法逆元。

实际上，我们可以证明，对于数 n 的这种特殊形式，知道 n 和 $\varphi(n)$ 可以得到因式分解 $n=p\cdot q$（需要很少的计算）。这个技巧的依据是，p 和 q 是如下方程的根：

$$x^2-(p+q)\cdot x+p\cdot q=0$$

知道 $p\cdot q=n$，因此我们可以用 n 和 $\varphi(n)$ 来表示 $p+q$，我们将得到这个用 n 和 $\varphi(n)$ 表示的方程的系数，容易分别给出 p 和 q。

因为有：

$$\varphi(n)=(p-1)(q-1)=p\cdot q-(p+q)+1=n-(p+q)+1$$

我们可以重新整理得到：

$$p+q=n-\varphi(n)+1$$

因此，p 和 q 为如下方程的根

$$x^2-(n-\varphi(n)+1)x+n=0$$

解得两个根 p 和 q 为：

$$\frac{-(n-\varphi(n)+1)\pm\sqrt{(n-\varphi(n)+1)^2-4n}}{2}$$

我们必须注意到，还有其他方法不需要分解 n 而得到明文，或者明文的一部分。

根据加密和解密指数 e 和 d 的知识似乎还不能得到素数分解 $n=p\cdot q$。因此，即使指数对 e 和 d 受到威胁，数 $n=p\cdot q$ 也是不可用的。但是，我们在后面将看到，私人（解密）密钥的暴露会危及整个密码系统。特别，有一个运行很快的拉斯维加斯算法将产生因式分解 $n=p\cdot q$。

由模数 n（两个秘密素数 p 和 q 的乘积）、公钥 e 和私钥 d 构成的密码体制，其用户并不需要知道素数 p 和 q。因此，**中心机构**（Central Agency）就有可能反复使用同样的模数 $n=p\cdot q$。但是，正如我们提醒的那样，危及一个密钥对将会威胁其他的密钥对。

加密/解密算法的计算速度。如果排除其他干扰，对大整数的大指数运算将花很长的时间。在 RSA 中加密和解密算法中都需要进行这种指数运算，因此从一个幼稚的观点来看，为什么执行算法本身比执行恶意攻击更容易这一点不是太明显。但是，实际上对于一些特定范围内

的数（一百位以上的数）来说，所需要的指数运算可以设计得比大整数的因式分解运算的速度快得多。即使是这样，到目前 RSA 的加密和解密算法（以及大多数非对称密码算法）的运算速度仍然比最好的对称加密算法要慢得多。

选择素数 p 和 q 的位数一般为一百多位。因此，尽管加密指数 e 选择得相对较小，可能是只有几位的十进制数，而 e 的乘法逆元（解密密钥）将和 n 差不多大小。因此，计算整数的大幂运算，并模大整数 n 的任务就必定在执行速度上比对 n 执行因式分解的任务要快。

下面我们将描述一个重要的初等快速幂运算。这个方法使得幂指数运算相对简单。这个算法在计算数的方幂或者其他代数元素中比较通用。也就是，在计算 x^e 时，我们并不是计算所有的 $x^1, x^2, x^3, x^4, x^5, \cdots, x^{e-1}, x^e$ 。

密钥生成和管理。根据秘密素数 p 和 q 生成一个模数 $n = p \cdot q$，从而决定一个密钥对 e 和 d，其中 $e \cdot d \equiv 1 \bmod \varphi(n)$，首先要求两个素数 p 和 q 至少满足例如 $> 10^{80}$。因为 RSA 的安全性取决于因式分解的困难性，比较幸运的是素性检测比因式分解为素数要容易得多。也就是说，尽管通常我们很难将大的数 $n = p \cdot q > 10^{160}$ 分解为素数（即使使用很好的算法），但是我们能够廉价得到许多大的素数 $p, q > 10^{80}$。

在选择了 p 和 q 之后，可以先选择解密（私人）密钥 d。因为只有 d 满足和 $(p-1)(q-1)$ 互素，才存在相应的加密密钥 e，而且欧几里得算法给出了一个有效的方法来计算 e。

得到 d 满足和 $(p-1)(q-1)$ 互素的方法之一就是如下所述的猜测和验证法。注意，因为 $p-1$ 和 $q-1$ 本身很大，我们不可能得到它们的素因数分解。我们随机选择一个大的素数 d，然后利用欧几里得算法找到 d 和 $(p-1)(q-1)$ 的最大公约数。如果这个最大公约数 >1，则继续寻找。因为 d 是一个大的随机素数，所以可能第一个猜测就已经满足与 $(p-1)(q-1)$ 互素的试探概率很高。我们将在后面对这一点以及相关知识进行扩展论述。

更多的技术备注还有：

- 有时加密指数选择为 3，而素数 p 和 q 不是模 3 余 1 的。
- 因为技术原因，有些人最近建议将 $2^{16}+1 = 65537$（为素数）作为加密指数，然后选择的 p 和 q 模 65537 不同余 1 的。
- $p-1$ 和 $q-1$ 都必须至少具有一个很大的素因数，因为如果 $p-1$ 或 $q-1$ 只有一个小的素因数，则可能存在对 $n = p \cdot q$ 的因式分解攻击（Pollard 的 $p-1$ 攻击）。
- 素数 p 和 q 最好不要相隔太近，因为在这种情况下存在对 n 的因式分解攻击（费马等）。
- 比率 p/q 最好不要太靠近一个具有较小的分子和分母的有理数，因为这样会使得 D. H. Lehmer 的对 $n = p \cdot q$ 的连续片断因式分解攻击可能成功。

我们将在后面论述各种不同的因式分解攻击方法。

关于攻击 RSA 的报导。目前，似乎对 RSA 的攻击只能发生在 RSA 没有得到**正确实现**的情况下，例如选择的模数太小。也就是说，当特定的**可避免的错误**发生时，似乎存在可利用的脆弱性。当然，具有讽刺意味的是，有些情况下只有在有人找到并进行攻击之后才发现某些选择是错误的。

当然，密钥的长度（也就是 RSA 的模数 $n = p \cdot q$ 的长度，其中 p 和 q 是秘密的素数）不能太短，否则对 n 进行强行分解的攻击所需的时间可能会比人们希望的要短。例如，在 1999 年的春季后期，Adi Shamir 设计了一个特殊的计算机“Twinkle”，部分是模拟的，部

分是数字的，它可以将成熟的因式分解攻击速度提高 100 或 1000 倍。它使得 512 位的 RSA 模数 N（大约 160 位十进制数，由两个 80 位的素数生成）对于高安全目的来说显得太小了。

而且，实际的问题通常并不是说密码具有完全的不可破解性，而是需要多长时间来攻破。攻击速度提高 1000 倍即意味着一个在 10 年内安全的密码算法现在只在 8 小时内是安全的。但是以前在 10^{30} 年内安全的密码算法仍然在 10^{27} 年内是安全的。

让我们简单回顾一下 RSA 算法的建立过程：秘密选择两个大的素数 p 和 q，且都与模 4 同余 3 的。**RSA 的模数** $N=p\cdot q$ 是公开的。公开选择一个加密指数 e，与 N 互素。通常选择 $e=3$，但是选择 $e=2^{16}+1=65537$ 可能更好。根据欧几里得算法很容易计算得到密码的解密指数 $d=e^{-1}\bmod(p-1)(q-1)$。将消息分解成固定长度的块 x，它可看作在范围 $1<x<N$ 内的整数。加密步骤为：

$$E_{N,e}(x)=x^e\%N$$

这一步可以由任何人完成，因为 e 和 N 是公开的。解密步骤为：

$$D_{N,d}(y)=y^d\%N$$

秘密的解密指数 d 被用来执行这个公式。(N,e) 为**公开密钥**，(N,d) 为**私有密钥**。

- 攻破 **RSA 函数**意味着在没有提前给出解密密钥 d 的情况下要求函数：$x\to x^e\%N$ 的反函数。也就是说，有人企图描述模 N 的 **e 次根函数**。
- 攻破 **RSA 密码**意味着以更高的概率攻破 RSA 函数，也即意味着在没有提供解密指数的情况下揭示明文或者明文的一部分。对于密码算法的不可破解性，而不是函数的不可破解性，通常被称为**语义上的安全**。
- 当然，如果知道因数分解 $n=p\cdot q$，则可以计算得到数 $(p-1)(q-1)$，从而可以找到解密指数 d，是描述 **e 次根函数的一种方法**。
- 目前似乎还不知道是否能够由模 N 的 e 次根得到 N 的因式分解，其中 $e\geqslant 3$。（相比之下，如果我们能够得到**模 N 的平方根**，则我们就有一个能够对 N 进行因式分解的较好概率算法。）
- 为了应用欧几里得算法来有效地用 $(p-1)(q-1)$ 由加密指数找到解密指数，则必须知道数 $(p-1)(q-1)$。同时知道 $Q=(p-1)(q-1)$ 和 $N=p\cdot q$ 使得人们可以通过解下面的二次方程得到素数 p,q：

$$x^2-(N-Q+1)x+N=0$$

因此**分解** N（采用高效的算法）等价于通过欧几里得算法计算解密幂指数 d。

- 因此对 RSA 的一个最明显的攻击就是企图分解模数 N。
- 根据 1999 年的标准，为了永久地防止因式分解攻击，N 应该为一个 1024 位的数，大约是 309 位的十进制数。这样 p 和 q 应为 512 位数，或者大约为 154 位的十进制数。
- 如果量子计算机得到实现，Peter Shor 在 1993 年的工作表明，存在一种快速的量子算法对大数进行分解，这就会改变很多事情！
- 另外，p 和 q 应该是**强素数**。

备注 “强素数”的概念似乎因人而异，不同的研究者有不同的说法，而且也取决于应用环境。第一个**强素数** p 的最简单的定义为：强素数 p 要满足 $p-1$ 能够被一个“大”素数整除。据说 $p-1$ 并不是**平滑的**。这个特性可抵抗 **Pollard 的 $p-1$ 攻击**（我们将在后面论述）。

对**强素数**定义的一个通常的改进是进一步要求 $p+1$ 也能够被大素数整除。也就是说，$p-1$ 和 $p+1$ 都不是**平滑的**。这也就抵抗了 Pollard 的 $p-1$ 攻击的一个变种攻击，即使用 $p+1$ 的小因子。对**强素数**的定义的进一步加强为：素数 p 使得 $p-1$ 可以被一个“大”素数 r 整除，$p+1$ 也能够被一个大素数整除，且 $r-1$ 也能够被一个大素数整除。如此循环下去。

- 到目前为止，还没有出现对 RSA 函数的直接因式分解攻击，假设素因子足够大而且是强素数。
- 如果出于一般的考虑，对于非常大的整数进行分解的最好算法是**数域筛法**，这种方法对于分解 n 位整数的渐进运行时间大约为：

$$e^{2n^{1/3}\log^{2/3} n}$$

作为 n 的函数，它是一个**子指数的**，而不是**多项式的时间估计**。Adi Shamir 最近（1999）发明的小工具“Twinkle”设计为在数域筛法中运行比在以前的通用计算机中运行要快得多。尽管 Shamir 的设备能够提高速度高达 1000 倍，但是它也没有产生质的变化，因为很容易通过使用更大的模数而使问题仍然回到了原来的分解问题。

前向搜索攻击。如果知道所有可能消息的集合，而且该集合相对来说比较小，则攻击者只需要将所有的消息进行加密直到找到一个匹配者。例如，如果已知消息为“Yes”或者“No”，则只需要计算一次加密就知道哪一个是明文。因此，特别是在比较短小的消息的情况下，应该通过在前面或者后面添加随机位从而对消息进行**填充**。

公共模数攻击。注意授权的解密者没有必要知道 RSA 模数的因式分解 $N=p\cdot q$。这就会导致在几个用户之间为了节省而试图使用同一个模数，而仅仅是解密（和解密）密钥不同。这是一个错误。已知 N 和加密/解密对 e 和 d 就可以对 N 进行因数分解，根据这个方法就可以得到任何其他的解密密钥。因此，一个 RSA 模数只能被一个人使用。

小解密指数攻击。为了节约计算时间，有人可能会设置参数使得解密指数相对较小（当然，不会小到使系统易受到穷举攻击的程度）。但是如果 $d<\frac{1}{3}N^{1/4}$，则存在有效的算法能够解出这个解密指数（见[Wiener 1990]）！这一论点使用了**连分式**的经典数论理论以及如何找到用有理数对二次无理数最合理的逼近方法。也就是说，对于 1024 位的模数，解密指数应该至少为 256 位，这对复杂的实现带来了更复杂的问题。后来，[D. Boneh, G. Durfee, preprint] 报告这个结果可以被“改进”到甚至对于更大的解密指数也可以进行快速的破解。在[Boneh 1999]中特别指出，使用小于 $\sqrt{N}$ 的解密模数的系统可能容易受到此类攻击。

小公开指数攻击。使用加密指数 $e=2^{16}+1=65537$（费马素数！）可以避免一些对 $e=3$ 的攻击。使用 $e=2^{16}+1=65537$ 意味着对于 1024 位的 RSA 模数加密要进行 17 次此种求幂运算，与之对比，对于 $e=3$ 只需要进行两次求幂运算，这很烦人。（17 次求幂运算是相对于“随机”加密指数的大约 1000 次求幂运算而言的。）但是，存在一些针对小加密指数的攻击。第一个就是：

假设 Alice 希望给几个不同的人发送同一个秘密消息。最简单的方法就是分别使用这些人的公钥对该消息进行加密，并发送出去。Eve 对 Alice 使用的非安全信道进行监听，他可以收集所有的加密信息。假设消息对于所有的接受者来说都在单个 RSA 块中。**如果消息的数目大于或等于加密指数 e，则 Eve 可以解出消息**。例如，如果 $e=3$ 且消息小到可以加密为单个 RSA 块，则只需要三个不同的加密就足够解出消息。

为了抵抗这种攻击，我们必须使用随机数来**填充**消息。这是一个很好的主意。当然，最简单的方法是使用简单的确定的填充，针对不同的接受者而不同。但是填充必须是随机的，否则这些填充毫无用处：

一些更加复杂的此类攻击为 **Hastad 的广播攻击**和 **Coppersmith 的短填充攻击**。但是如果使用更大的加密指数 $e = 2^{16} + 1 = 65537$，则这个方向上的脆弱性似乎可从本质上消除。

而且，当（公开的）加密指数很小时，解密密钥的**部分暴露**可以完全破解 RSA 密码算法，见参考文献[Boneh 1999]。在[Boneh, Durfee, Frankel 1998]中，给定解密（私人）密钥 d 的 $n/4$ 个重要的位，就可以在 $e\log_2 e$ 的线性时间内解出整个加密密钥，其中 e 为加密密钥。这与一个因式分解结论十分相关：**给定 *P* 的 *n/4* 个高位比特或 *n/4* 个低位比特，我们可以对 *N* 进行有效的分解**[Coppersmith 1998]。

实现攻击。特别是在使用智能卡的情况下，攻击者在实际上可能具有无限的机会进行已知明文攻击，RSA 的早期形式易遭受时间攻击，攻击者可以测量计算需要的时间，进而寻找在解密过程中的求幂运算的数目，从而获得解密指数中的信息[Kocher 1996]。

习题

10.2.01 一个 RSA 密码的模数为 12091，加密密钥为 3。明文为“2107”（我们并不关心将明文文本转化为整数的方式）。请对其加密。

10.2.02 一个 RSA 密码的模数为 210757，加密密钥为 3。明文为“12345”（我们并不关心将明文文本转化为整数的方式）。请对其加密。

10.2.03 一个 RSA 密码的模数为 210757，加密密钥为 3。明文为“54321”（我们并不关心将明文文本转化为整数的方式）。请对其加密。

10.2.04 一个 RSA 密码的模数为 12091，加密密钥为 3。请计算解密密钥。

10.2.05 一个 RSA 密码的模数为 14659，加密密钥为 3。请计算解密密钥。

10.2.06 一个 RSA 密码的模数为 15943，加密密钥为 3。请计算解密密钥。

10.2.07 一个 RSA 密码的模数为 1019×1031，加密密钥为 3。请计算解密密钥。

10.2.08 一个 RSA 密码的模数为 12091，加密密钥为 3。密文为“9812”。请对其解密（只对整数操作，我们假设已经以某种未知的且不相关的方法对文本进行了编码）。

10.3 Diffie-Hellman 密钥交换

本算法既是最初的公钥密码思想，又是当前人们使用的一种重要的机制。其实用之处在于对称密钥算法通常比非对称密钥算法速度要快得多（不论是硬件实现还是软件实现），因此公钥密码算法仅仅被用来建立私人密钥，即只在一次会话中使用的**会话密钥**。

这个协议使用一些特殊的技术，包括离散对数，当然也利用了计算离散对数的相对困难性。因此，它不仅仅工作于 $\mathbf{Z}/p$ 上离散对数的情况，而且可以被扩展到任意有限域、椭圆曲线或者可以使用对数的其他群结构。

首先，Alice 和 Bob 协商一个大的素数 m，以及一个模 m 的本原根 g。这些数不需要保密，且可以被一群用户共享。Alice 选择一个大的随机整数 x，秘密计算 $X = g^x \% m$，并通过尽可能安全的信道将 X 发送给 Bob。与此同时，Bob 也选择一个大的随机整数 y，秘密计算 $Y = g^y \% m$，并通过尽可能安全的信道将 Y 发送给 Alice。

于是 Alice 秘密计算 $k = Y^x \%m$，而 Bob 也同样计算 $k' = X^y \%m$。实际上有，$k = k' \bmod m$，证明如下：进行模 m 的计算，我们可以得到：

$$k' = X^y = (g^x)^y = g^{xy} = (g^y)^x = k \bmod m$$

因为模 m 的整数可充分利用指数的性质。但是网络上的其他任何人都不可能获得 k，除非他们能够计算离散对数，那么密钥 k（$=k'$）就可以直接或者间接用作对称密码的密钥。

为了安全起见，素数模 m 必须非常大，而且 m 最好是一个强素数，即最重要的是 $m-1$ 应该可以被一个相对大的素数整除。我们可以想像，m 最好是一个 **Sophie Germain 素数**，即(m – 1)/2 也是素数。但是所有素数中这种素数十分稀少。它们和**孪生素数**一样稀少，即素数 p 使得 p + 2 也是素数。目前人们还不知道是否存在无限多个 Sophie Germain 素数（也不知道是否有无限多个孪生素数）。

实际上，协议并不要求 g 必须为本原根，只要求（因为安全原因）它生成一个大的子群 $\mathbf{Z}/n^x$，因此攻击者就不能计算出相关的“对数”。

和许多其他类似协议一样，这个协议的最初形式也易遭受敌人主动的**中间人攻击**：监听者可以主动截取 Alice 发给 Bob 或者 Bob 发给 Alice 的消息，而且可以使用他们自己的消息来替换这些消息，从而可以分别与 Alice 和 Bob 单独（有效地）执行 Diffie-Hellman 交换协议，并使得 Alice 和 Bob 误以为他们还是在直接进行通信。如果 Alice 和 Bob 实际上没有进行事先联系，从而因此没有其他办法来相互验证对方身份，那么就很难防止各种形式的伪造攻击。

公钥合伙人（Public Key Partners）对这个协议及其改进版本申请了专利，但是该专利的有效时间截至 1997 年 4 月 29 号。

10.4 ElGamal 密码

这个思想来源于[ElGamal 1985]。该算法比 RSA 算法稍微复杂一些，但仍然是一个十分基本的算法。与 RSA 相比较，ElGamal 密码算法的思想在技术上的推广性更加现实。例如，椭圆曲线密码系统与 ElGamal 密码类似。

困难的任务。这里的困难任务是**计算离散对数**。具体如下，给定一个模数 m 和整数 b、c。一个满足下列等式的整数 x 就是 c **模** m 且以 b 为底的**(离散)对数**：

$$b^x \equiv c \bmod m$$

对于随机数 m、b 和 c，可能不存在这样的 x，知道这一点很重要。但是对于素数模 p 和一个适当选择的底 b 而言，对于任意不能被 p 整除的 c，将存在一个离散对数（这个理论将在后面阐述）。

给定一个大素数 $p > 10^{150}$。对于两个整数 b、c，假设我们知道对于某个 x，满足：

$$b^x = c \bmod p$$

困难任务就是在给定 b、c 和 p，计算 x。

对于任意正整数 m，如果每一个与 p 互素的整数 c 都可以表示为如下形式，

$$c = b^x \bmod m$$

则整数 b 通常被称为**模** p 的**本原根**。我们将在后面看到，对于特殊的整数才存在模 m 的原根，主要包括素数模。对于素数模 p，我们可以看到一个模 p 的本原根 b 具有下列特性：b 方幂 b^k 中满足模 p 同余 1 的最小正整数幂次为 $p-1$。也就是，

$$b^{p-1} \equiv 1 \bmod p$$

且没有更小的方幂满足这一条件。更一般的情况，对于与可被素数 m 整除的模数互素的任意 x，x 模 m 的**阶**或**指数**为满足下式的最小的正指数 n：

$$x^n \equiv 1 \bmod m$$

我们可以看到本原根具有最大的阶。

对于 ElGamal 密码，严格地讲并不是必须具有一个模 p 的本原根，因为我们只需要构造 $b^l \equiv c \bmod p$，但是我们必须要求 b 模 p 的阶尽可能的大，否则该密码就太容易被破解了。

“离散对数”的思想和利用计算离散对数的困难性构建一个密码体制的作法，都能够做进一步的抽象。这个结论最流行的例子就是**椭圆曲线密码**，要描述该算法需要大量的准备工作，我们将在后面进行论述。

加密和解密的描述。给定一个大素数 $p > 10^{150}$，一个模 p 的本原根 b（意味着任何数 y 都可以表示为：$y = b^L \bmod p$），以及一个在范围 $1 < c < p$ 内的整数 c。密钥就是使得 $b^l = c \bmod p$ 的方幂 l（**离散对数**）。只有解密者知道 l。

加密步骤如下：加密者知道 b, c, p。明文 x 被编码为一个整数，在范围 $0 < x < p$ 之间。加密者选择一个辅助随机整数 r，这是一个只有加密者知道的临时秘密数，并对明文 x 加密如下：

$$y = E_{b,c,p,r}(x) = (x \times c^r)\%p$$

加密者将这个加密的消息与“头部” b^r 一起发送。注意加密者只需要知道 b, c, p 以及随机选择的整数 r，但不知道离散对数 l。

解密步骤需要离散对数 l 的知识，而不是随机整数 r。解密者知道 b, c, p, l。首先，从“头部” b^r 中解密者可以计算得到：

$$(b^r)^l = b^{r \cdot l} = (b^l)^r = c^r \bmod p$$

则可以通过乘以 c^r 模 p 的乘法逆元 $(c^r)^{-1}$ 解出明文：

$$D_{b,c,p,r,l}(y) = (c^r)^{-1} \cdot y\%p = (c^r)^{-1} \cdot c^r \cdot x \bmod p = x\%p$$

备注 实际上，b 没有必要是原根，但是必须满足的条件：满足 $b^k = 1 \bmod p$ 的最小正整数 k 几乎和 p 一模一样大。我们可以将这种 b 称为**准本原根**（拟本原根）。

因此，Alice 为了建立起一个离散对数公钥密码系统，从而使得其他人可以向她发送只有她才能解密的加密消息，她选择一个大的随机素数 p，选择一个随机的模 p 的准本原根 b，在范围 $1 < l < p-1$ 中选择一个随机整数 l，并计算：

$$c = b^l \bmod p$$

因此，并不是实际计算一个离散对数，Alice 首先选择“对数” l，然后计算数 c，c 以 l 作为其对数。她对 l 保密（l 位私人/秘密密钥），但是公开 b, c, p。当任何其他人想要给她发送消息时，他们就拥有足够的信息，即 (b, c, p)，来进行加密运算。但是只有 Alice 才能够解密，因为只有她才知道该离散对数。

例子 我们可以举一个小例子来说明这个过程。Alice 选择 $p = 1009$ 作为她的“随机”素数，以及 $b = 101$ 作为一个随机的准本原根。她选择“随机”对数 $l = 237$，并计算：

$$101^{237}\%1009 = 482$$

她公布 $(b, c, p) = (101, 482, 1009)$，并保持 $l = 237$ 为秘密信息。当 Bob（或其他任何人）想要向 Alice 发送消息 $x = 559$，他就执行如下过程。他选择“随机”的 $r = 291$（他最好在不同的场

合下选择不同的r），并计算：

$$c^r = 482^{291}\%1009 = 378$$

然后 Bob 给 Alice 发送：

$$x \cdot c^r\%1009 = 559 \cdot 378\%1009 = 421$$

以及头部

$$b^r\%1009 = 101^{291}\%1009 = 661$$

当 Alice 接收到消息/头部对(421,661)后，她解密如下。因为 Alice 知道离散对数l，根据$b^r = 661$她可以计算得到c^r（所有运算都模 1009）：

$$c^r = (b^l)^r = (b^r)^l = 661^{237} = 378 \bmod 1009$$

Alice 于是计算乘法逆元：

$$378^{-1} \bmod 1009 = 670$$

然后，她最后计算明文：

$$\text{明文} = 378^{-1} \cdot 421 \bmod 1009 = 670 \cdot 421 = 559$$

这样，她就解出了原始消息“559”。

备注 注意 Alice（和 Bob）必须能够计算整数模 1009 的较大方幂。存在一个很好的算法来实现这个目的，我们将在稍后论述。

ElGamal 密码算法安全性的基本方面。这个密码算法的安全性取决于计算整数c**基于**b**的离散对数**l的难度，且模素数p，而且这个整数满足：

$$b^l = c \bmod p$$

这些对数与实数或复数的对数之间几乎不存在某些切实的联系，尽管他们共有一些抽象的特征。计算离散对数的早期算法是进行简单的试验。计算离散对数的更好算法（例如，攻击 ElGamal 密码算法）要求对模p的整数有更多的了解。

为了避免在一些特殊情况下一些特殊的**对数计算攻击**，我们必须选择p使得$p-1$不具有“太多”的小素数因子。因为$p-1$是偶数，它总是具有因子 2，但是除了这个因子，我们希望能够避免其他小因子。

加密/解密算法的速度。和 RSA 一样，加密和解密算法的速度主要取决于模n的指数运算的速度，后者可以得到适当的提高。ElGamal 算法（以及相关算法）具有这样一个特征，即加密者需要一个很好的随机数供应源。因此，获得高质量的伪随机数生成器对于 ElGamal 算法很重要。

对于密钥生成和管理的建议。对这个密码系统的最明显要求就是普通的大素数p的选择，要求大约$p > 10^{160}$。试除法对于这个要求是完全不够的。无论谁构造$b^l \equiv c \bmod p$大概都将首先选择一个大的素数p，很有可能还要满足额外的条件。对于这种或其他任何目的考虑，一种特殊的好素数具有形式$p = 2 \cdot p' + 1$，其中p'为另一个素数。在这种情况下，在范围$1 < b < P-1$内的数b大约有一半为本原根（因此具有阶$2p'$），而另一半具有阶p'，它们也不算太差。因此，b的随机选择就可能有许多很好的候选者。随机指数l的选择以及$c = b^l\%p$的计算完成之后，准备工作也就完成了。

如果素数p具有特殊形式$p = 2 \cdot p' + 1$，而p'为素数，很容易找到本原根，因为（我们将在后面理解）阶为p'（而不是$2p'$，本原根将具有阶$2p'$）的元素是$\mathbf{Z}/p$中所有的**平方(或二次幂)**。是否为一个平方的特点很容易通过使用二次符号计算得出，和我们在后面将看到的

一模一样。因此，对于素数 $p=2\cdot p'+1$，要求在 ElGamal 算法中使用的数 b 为模 p 的本原根是可行的。

对于几个关键构造 $b^l=c \bmod p$，使用单个素数模 p 仍有点似是而非，因为模 p 可能有许多不同的本原根，所以一旦用其中一个本原根所做的构造受到攻击，则将会危及其他的构造。

另外，任何加密者在加密过程中将需要一个伪随机整数的良好供应源。这本身也是一个问题。

习题

10.4.01 对于 ElGamal 密码算法，给定公开信息 $b=2$，$c=58$，$p=103$，请使用 $r=31$ 作为辅助“随机”数，对消息“87”进行加密。（假设“87”是一个文本的数字编码。）

10.4.02 验证 $c=58$ 以 $b=2$ 为底模 $p=103$ 的离散对数（与前一个问题相关）为 $l=47$。

10.4.03 对于 ElGamal 密码算法，给定公开信息 $b=2$，$c=58$，$p=103$，使用私钥 $l=47$，对密文“79”进行解密。

10.5 Knapsack 密码

不存在单个的“Knapsack 密码”，而是存在使用同样数学问题的一个密码族，这个问题即指 knapsack 问题或者子集和问题。Merkle 和 Hellman[Merkle，Hellman 1978]首先发明了这类密码。Adi Shamir[Shamir 1982]找到了一些确定性的攻击方法。实际上对 Knapsack 密码的攻击和企图破解它们的进展代表了一种反复的过程，通过这种过程产生了实际的密码。我们在这里将只描述基本的形式，并简要指出 Shamir 攻击的本质。不幸的是，不同的 knapsack 密码被发现有问题的次数太多以至于有些人对它们失去了热情，尽管大多数改进形式似乎没有问题。

Knapsack 问题为：给定一系列正整数 $a_1,\cdots,a_n$，并给定另一个整数 b，如果可能的话，找到一个子集 $a_{i_1},\cdots,a_{i_k}$（每一个元素 a_i 最多只能使用一次，尽管其中一些 a_i 的值可能一样）满足：

$$a_{i_1}+\cdots+a_{i_k}=b$$

关于这个问题有两种不同的论述：其中一个比较简单，即给定 a_i 和 b，是否存在一种解决方案。另一个则承认存在一种解决方案，请找到它。所有这些问题都（可能！）是 NP 完全问题，意味着要解决这些问题几乎很困难，但是很容易验证一个解决方案的正确性。当然，如果大约 $n=300$，一个针对 a_i 的所有 2^n 个子集的穷举搜索攻击需要很长时间，如果 $n=4096$，则更长。

为了讨论的方便，我们称 $a=(a_1,\cdots,a_n)$ 为 **knapsack 向量**，而 b 为 knapsack 的**大小**。在一个解 $b=a_{i_1}+\cdots+a_{i_k}$ 中出现的 a_i 存在于**大小**为 b 的 **knapsack** 中。一个长度为 n 的 knapsack 向量 $a=(a_1,\cdots,a_n)$ 可以被用来对由数位（0 和 1）组成的向量 $x=(x_1,\cdots,x_n)$ 进行加密，通过计算一个结构上像点积的函数：

$$b=f_a(x)=x_1a_1+\cdots+x_na_n$$

因此，这个数量只计算了那些 a_i 中对应的 x_i 为 1 的数。然后发送 knapsack 向量 a 和大小 b。解密为寻找加起来等于大小 b 的 knapsack 向量元素的子集。因此，解密步骤等于解决一个

knapsack 问题。我们注意到，解决一个 knapsack 问题通常来说很难。因此我们处于一种荒谬的境地，即授权的和非授权的解密通常都很困难。

当然，人们可能马上会想到一个问题，即是否可能存在多于一个的解密方案，这是一个坏事情。因此一个具有下列特性的 knapsack 向量被称为**单射**，即对于给定的大小，最多存在一个 knapsack 问题的解决方案。我们可将上面提到的函数

$$f_a(x) = x_1a_1 + \cdots + x_na_n$$

作为对 n 位字符串 x 可能的解密函数。knapsack 向量 $a = (a_1, \cdots, a_n)$ 是单射当且仅当 f_a 为单射。特别地，这要求所有的 a_i 都不同。

根据一般 knapsack 问题的 NP 完全性，我们对于上面所述密码的非授权解密的难度可以很自信，但是授权解密者也具有同样的难度。为了使得授权解密者能够解决一个相对容易的问题（进行解密），具体计划是进行一定的安排使得授权解密者能够解决一个更容易的子问题，现在我们对此进行论述。

如果向量中每一个 a_i 都是严格比前面一个元素的值要大，则一个 knapsack 向量 $a = (a_1, \cdots, a_n)$ 是**超增长的**，也就是说，对于每一个下标 i，有：

$$a_1 + \cdots + a_{i-1} < a_i$$

如果 knapsack 向量 $a = (a_1, \cdots, a_n)$ 是超增长的，那么就存在一个更加简单的方法来解决大小为 b 的 knapsack 问题：

- 如果 $b < a_n$，则 a_n 不能出现于大小为 b 的 knapsack 中，因为它不匹配。
- 如果 $b \geq a_n$，则 a_n 必须出现于大小为 b 的 knapsack 中，因为根据超增长特性，其他的 a_i 累加起来比 a_n 要小，因此显然不能累加到 b。

在第一种情况中，保持同样的大小 b。在第二种情况下，用 $b - a_n$ 代替 b。在每种情况中，都用 $(a_1, \cdots, a_{n-1})$ 代替 knapsack 向量。这样就把问题转化为一个具有更短的 knapsack 向量的新 knapsack 问题。很明显，当我们达到 a_1 时，处理将终止，乐观地说，如果 $a_1 = b$，则问题将会成功解决（那时 b 具有新的值），如果 $a_1 \neq b$，则结果是否定的。这也表明对于具有超增长 knapsack 向量的 knapsack 问题来说，最多只存在一种解决办法。因此，如果授权解密者可以通过解决超增长 knapsack 问题而进行解密，那么这种解密才被认为是容易的。

但是如果 knapsack 问题是明显超增长的，非授权解密者就能够做同样的事情。因此，我们认为，将超增长特性伪装成为一种只有授权解密者知道的行为，而不是为攻击者知道的行为。尽管 knapsack 项目列表被进行重新整理，如果做得够聪明的话，对 m 件事情的排序将只需要 $O(n \log n)$ 个步骤。

隐藏一个简单问题的第一个思想是通过一个秘密的模某数的乘法和减法。Alice 选择一个超增长的 knapsack 向量 $(a_1, \cdots, a_n)$，以及一个整数（**模数**）：

$$m > a_1 + \cdots + a_n$$

选择 m 比所有 a_i 的总和还要大，这使得这个程序有时被称为**强模乘法**。她选择另一个与 m 互素的数（**乘数**）t，因此 t 具有一个模 m 的乘法逆元。然后 Alice 计算：

$$c_i = (ta_i)\%m$$

这样得到一个新的 knapsack 向量 $c = (c_1, \cdots, c_n)$，Alice 将其公布给加密者作为她的公钥。Alice 秘密保存 t 和 m（以及 t 的模 m 的乘法逆元值）。如果 Bob 要加密一个只有 Alice 才能阅读的消息，则加密步骤与前面给出的类似，使用一个更改的 knapsack 向量 c 来加密一个 n 位的消

息 $x=(x_1,\cdots,x_n)$ 如下：

$$b=f_c(x)=c_1x_1+\cdots+c_nx_n$$

Bob 将密文 b 发送给 Alice。Alice 的解密步骤为，首先计算：

$$t^{-1}b\%m=t^{-1}(c_1x_1+\cdots+c_nx_n)$$

$$=(t^{-1}c_1)x_1+\cdots+(t^{-1}c_n)x_n=a_1x_1+\cdots+a_nx_n \bmod m$$

因为 m 比所有 a_i 的总和还要大，这个模 m 的等式实际上是一个整数的等式，意味着：

$$(t^{-1}b)\%m=a_1x_1+\cdots+a_nx_n$$

因为 Alice 知道 a_i 的超增长 knapsack 向量，否则可以在任何情况下根据 c_i 计算得到它们：

$$a_i=t^{-1}\cdot b_i\%m$$

她可以很容易解决 knapsack 问题并如上所述方法解密。因此似乎我们得到了一个似是而非的公钥密码。法定的接收者拥有比通常的 knapsack 问题更加容易解决的问题，因此如果一个敌人企图对通常的 knapsack 问题进行敌意的解密行为，则将会很难。

问题 敌人为了将问题转化为一个超增长的 knapsack 问题并不需要找到秘密的 t 和 m（因此敌人并不需要解决一个通常的 knapsack 问题）。总而言之，如果敌人足够幸运地找到了任意的 t' 和 m' 使得：

$$a'=(t'^{-1}c_1\%m',\cdots,t'^{-1}c_n\%m')$$

为超增长的，这就将此问题转化为一个相对容易的问题。按照这个思想，Adi Shamir[Shamir 1978] 发明了一种方法，可以在多项式时间内找到一对 (t',m') 以将公布的向量转化为一个超增长向量。这就破解了这个最简单的 knapsack 密码。

备注 还要注意的是，此密码算法很容易遭受选择明文攻击。

10.6 NTRU 密码

1995 年由 J. Hoffstein, J. Pipher 和 J. Silverman[Hoffstein, Pipher, Silverman 1996]发明的 NTRU 密码是一个在数学上比 RSA 或 ElGamal 密码还要复杂的密码，而且因为其复杂性和出现的时间相对较短，所以还不能说已经对其进行了完全彻底的研究。毕竟，将整数分解成为素数的问题是非常古老的，而且特别是自从计算机诞生以来计算机科学家们已经对其进行了很久的研究，而需要用来理解或攻击 NTRU 密码的计算类型仅仅从 19 世纪末闵可夫斯基开展相关工作以来才日显其重要性，而且计算机科学家们才对其研究了仅仅 25 年。但是认为新事物的发明不可能比旧事物要好的观点是愚蠢的。NTRU 密码也申请了专利。

NTRU 密码的一个很重要的特性就是和 RSA 或者椭圆曲线密码算法（“ECC”）相比较，很明显它在某些方面具有很大的优点，但是在其他一些方面又具有缺点。从正面来说，（具有明显的安全性）NTRU 算法比 RSA 或 ECC 算法运行的速度要快许多倍，其密钥生成也更快，且需要更少的存储空间。从负面来说，与 RSA 或 ECC 相比，NTRU 算法需要更大的带宽和更大的密钥空间。例如，对应于 1024 位安全性级别的 RSA，NTRU 的公钥长度为其两倍。但是这种差异会随着安全性级别的增大而降低。如果或者当 NTRU 密码算法和老的密码算法得到同样信任，这些性能平衡将会在实际应用中给 NTRU 算法带来很多重要的优点。

另一个重要的特征就是，（目前据我们所知）只有当量子计算机得到应用时，NTRU 密码算法才有可能被破解，而 RSA 和普通的 ElGamal 一定已经被破解了。自从 1993 年 Peter Shor[Shor 1996a]发明针对大整数的快速因数分解量子算法以来，对于 RSA 或普通的离散对

数密码算法来说，这种可能性对将来构成威胁。当然，针对其他公钥密码算法的“困难问题”可能也将发明快速量子算法，但是到那时 RSA 和 ElGamal 密码算法已经被称为最易破解的算法了。

关于 NTRU 密码的建立和运行的描述，与 RSA 或 ElGamal 密码算法相比更简单，尽管它只会用到一些非常基本的抽象代数和数论理论。与之对比，理解椭圆曲线离散对数密码所需要的数学复杂性则要大的多。

困难问题。假设是困难的计算任务，并因此假设是 NTRU 密码安全保证，因为它不像素数分解或离散对数那么初等，故我们不会对其进行描述，。一个必须保持是困难的任务就是在格中寻找到最小的向量。（对于目前的讨论，一个“格”就是向量的一个“好的”集合，有可能存在于一个高维空间中。）这里有一个比较好的算法，即 LLL（Lenstra-Lenstra-Lovasz）算法[Lenstra, Lenstra, Lovasz 1982]，[Schnorr, Euchner 1994]对其进行了改进，后者能够在典型的格中找到一个短向量。但是当最小的向量接近于或者长于最短向量大小的“期望值”时，LLL 算法就不能很好地执行了。NTRU 的参数选择必须使其利用这个效果。

加密和解密的描述。给定正整数参数 N,p,q，其中 p,q 不必为素数，但是它们之间必须互素，且满足 $gcd(N,pq)=1$。设 R 为关于 x 的多项式的集合，具有整数系数和严格比 N 小的次数，并且具有似乎比较特殊的乘法★：

$$x^i \star x^j = x^{i+j\%N}$$

也就是：

$$\left(\sum_{0\le i<N} a_i x^i\right) \star \left(\sum_{0\le i<N} b_i x^j\right) = \sum_{i,j} a_i b_j x^{i+j\%N}$$

$$= \sum_{0\le k<N} \left(\sum_{i+j=k\%N} a_i b_k\right) x^k$$

R 中的加法为通常的多项式加法。从这个描述看出，这个乘法是否满足交换律、结合律以及加法的分配律都还不是很明确，尽管它具有所有这些特性。

我们有关于 R 的一些特殊运算，称为模 p 约简和模 q 约简，意味着分别对多项式的系数做模 p 或 q 的约简。与我们通常的应用相比较，我们可以写作：

$$f\%p = \text{系数经过模 } p \text{ 约简的多项式 } f$$

$$f\%q = \text{系数经过模 } q \text{ 约简的多项式 } f$$

注意这两个约简的输出之间没有任何联系。如果有另一个多项式 F 满足下列等式，则我们说多项式 f 具有一个**模 p 的逆元**：

$$(f \star F)\%p = 1$$

为了创建一个 NTRU 密钥，Alice 选择两个次数为 $N-1$ 的多项式 f 和 g，确保 f 具有一个模 p 的逆元 F_p 和一个模 q 的逆元 F_q。Alice 的公钥为：

$$h = (F_q \star g)\%q$$

私钥为 f。

对于这个密码，消息必须为次数为 $N-1$ 的多项式，具有模 p 约简的系数（根据我们目前的理解，它们存在于范围内 $(-p/2, p/2]$。）如果 Bob 要为 Alice 加密一个消息 x，它随机选择一个次数为 $N-1$ 的多项式 φ 并计算：

$$y=(p\varphi \star h+x)\%q$$

然后将其发送给 Alice。

尽管可以通过一个合适的概率算法来进行系统的改进，解密还是可能会失败。Alice 要解密消息 y，她需要根据前面 f 计算得到的多项式 F_p（可以存储在一个安全的地方）。Alice 计算：

$$a=(f \star y)\%q$$

然后，Alice 通过下列计算进行解密：

$$x=(F_p \star a)\%p$$

备注 从上述描述中，我们还不是很清楚，上面所声称的解密方法是否真正能够解出原始消息。实际上，应该正确选择参数以便能够以很高的概率完成解密任务。

备注 根据我们后面将会论述的环理论，具有普通加法和特殊乘法的多项式集合 R，是商环构造的一个标准例子。特别是，简单定义 R 为：

$$R=\mathbf{Z}[x]/(x^N-1)$$

类似的对象经常在各种不同的数学、科学或工程领域中的“卷积”计算中使用，但在这些应用中通常没有明确提出商环的思想。这个观点的一个实质就是它提供了一个系统方法来理解“乘法”运算，例如上面介绍的确实能够正常进行。

为什么解密能够进行？ Alice 在解密步骤中计算的多项式为：

$$\begin{aligned}
a &= f \star y \\
&= f \star (p\phi \star h+m)\%q && \text{(根据加密的定义)} \\
&= f \star (p\phi \star F_q \star g+m)\%q && \text{（根据}h\text{的构造）} \\
&= f \star p\phi \star F_q \star g+f \star m)\%q && \text{（根据分配律）} \\
&= (f \star F_q \star p\phi \star g+f \star m)\%q && \text{（根据交换律）} \\
&= (1 \star p\phi \star g+f \star m)\%q && \text{（根据}F_q\text{模}q\text{的可逆特性）} \\
&= (p\phi \star g+f \star m)\%q
\end{aligned}$$

然后，如果需要使得所有的系数都在范围 $(-R/2,q/2]$ 内，则 Alice 可以通过减去 q 来调整系数。通过对参数的仔细选择，即使在模 q 的约简运算之前，也可以安排所有的系数都在范围 $(-R/2,q/2]$ 内，因此模 q 的约简不会造成信息的损失：在那种情况下，当（目前的正规版本）a 为模 q 约简的时候，系数将不会改变。因此，模 q 约简步骤没有做任何事情，Alice 就准备计算：

$$\begin{aligned}
(a\%p) &= (p\phi \star g+f \star m)\%q\%p \\
&= (p\phi \star g+f \star m)\%p \\
&= 0 \star g+f \star h\%p && \text{（因为}p\phi\%p=0\text{）} \\
&= f \star m\%p
\end{aligned}$$

则通过星号乘法乘以 F_p 可以解出明文：

$$\begin{aligned}
(f \star m) \star F_p\%p &= (f \star F_p) \star m\%p \\
&= 1 \star m \\
&= m
\end{aligned}$$

其中，我们又一次使用运算★的交换律和结合律，以及定义 F_p 的特性：

$$f \star F_p \% p = 1$$

参数设置。实际上，算法的正式描述要求/建议，f, g 和 ϕ 总是被选择为只具有 ±1 和 0 的系数。当然，还有进一步的基于部分启发式或部分理论考虑的特殊建议。和通常的密钥空间的考虑不同，应该选择适当的参数以抵抗格攻击，这种攻击将在下面进行讨论。

安全性。第一点在于，通过得到实际密钥空间的大小的平方根，相当标准的中间人攻击（利用生日悖论的思想）可以极大减小搜索空间。因此，要求攻击者搜索 10^{100} 个可能性就必须得从具有 10^{200} 个元素的集合中选择 f, g, ϕ。

针对 NTRU 密码的最严重的一类攻击似乎是所谓的格攻击，这种方法将密钥 f 或多或少看作一个特殊的向量集合（“格”）中最小的向量。LLL 算法可以很快找到一个短向量，除非通过使得最短向量的长度接近于甚至长于这种长度（对于合适的“随机”格）的期望值，才能防止这种攻击。然而，为了破解 NTRU 密码，LLL 算法不得不在格中找到其中一个最短的向量，其中所有的“短”向量最好都具有中等长度，而不是很短。在 NTRU 密码中，合适的参数设置似乎能够阻止 LLL 算法的攻击：试验证据表明，运行时间可以是参数 N 的超指数。特别地，这也是建议相对小的 N 值可以使得 NTRU 安全地实现。

注意到 NTRU 中的多项式乘法常是具有系数为 1，0 以及 −1 的多项式乘法，并且参数 p 和 q 时“正常的”整数，根据密码学标准来说是较小的数，因此所有的操作都是针对于小/中等的整数，而不是特别大的整数。这一点有利于提高 NTRU 密码的速度。

备注 NTRU 的一个不寻常的特性是对于每一位明文，它发送大约 $\frac{\ln q}{\ln p}$ 位密文。因为必须满足 $q > p$ 以避免丢失信息，这就表明了 NTRU 算法与老的密码相比是多么浪费带宽。

当我们拥有量子计算机时，能够提高格攻击的速度，而眼下(2000 年中期)还没有量子算法。当然这并不排除在将来能够发明此类算法。但是，与 RSA 的情况以及素数分解形成强烈的对比：Shor 的针对因式分解的量子算法可以完全破解 RSA。

对于相关的加密/解密速度、明显的安全性、以及更多的细节讨论，可以参见 NTRU 的主页：http://www.ntru.com / 。

10.7 算术密钥交换

这里谈到的密钥交换机制非常抽象，因此对于第一次阅读这部分内容的读者可能应当略过。但是，它却表明了现代数学在密码学中所起的重要作用。

算术密钥交换[Anschel, Anschel, Goldfeld 1999]是一个新的密钥交换机制。它的目标是为了实现和上面讨论的 Diffie-Hellman 密钥交换同样的功能，也就是当必须通过一个不安全的信道进行通信时建立一个共享密钥。与 Diffie-Hellman 相比较，它建立在一些复杂的数学问题基础之上，并且给出了似乎合理的安全性声明。但是，[Hughes, Tannenbaum 2000]的长度攻击给出了一个令人信服的启发式的结论，即似乎可以破解该密码（以及相关密码）。密钥交换和目前已提出的攻击都很难用实际的术语来解释，但是考察密码算法和它们的机制时提到这些可能性似乎是一个很好的想法。

对密钥交换的描述需要一些群论的思想，我们会在后面系统地讨论这方面的内容，但是现在可以略过。一些关于使用群论来构造密码的一般思想早先出现于[Wagner, Magyarik 1985]。在[Abschel, Anschel, Goldfeld 1999]中的一个重要的新思想就是辫群的使用，或者更加

一般的阿廷群，这些思想都使用了[Birman, Ko, Lee 1998]中发明的一种新算法。代数密钥交换申请了专利。

设 G 为一个**群**。简单地（见后面章节）说，这意味着 G 为一个集合，具有一个元素 $x, y \in G$ 的二元运算，记做 xy，通常该运算为普通的乘法运算。很重要的一点是我们不能假设该运算是可交换的。也就是说，可能有 $xy \neq yx$。群中也有一个单位元 $e \in G$，对于所有 $g \in G$ 具有特性：$eg = ge = g$。此外群中的元素存在逆元。也就是，对于每一个 $g \in G$，存在一个 $g^{-1} \in G$，使得：

$$gg^{-1} = g^{-1}g = e$$

最后，我们要求该运算满足结合律，即：

$$(xy)z = x(yz)$$

这是一个相当抽象的概念，而且对于所有这种一般思想的分支将会是怎样的，并不是完全清楚。实际上，“群”的概念花费了 100 多年才发展到现在的形式，大约是从 1800 年到 1920 年。

作为“群”的一个例子，具有实数元素的非零 2×2 矩阵的集合满足这些公理，其单位元为：

$$e = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

我们可能记得，矩阵乘法通常不满足交换律，可以用很简单的例子来证明。

给定 G 中一个固定集合 $S = \{s_1, \cdots, s_n\}$，S 中的一个**字**为分类 $s_{i_1}^{k_1} s_{i_2}^{k_2} \cdots s_{i_N}^{k_N}$ 的任意表达式，其中指数 k_j 为正整数或负整数。例如：

$$s_1 s_2^{3} s_1^{-7} s_2$$

是以 s_1, s_2 以及它们的逆构成的字。或者

$$aabbaba^3b^{-1}$$

为 a, b 以及它们的逆构成的字。“字”实际上就是通过元素及其逆重复相乘得到的任意表达式。如果 G 中的每一个元素都可以用 S 中的元素及其逆表达为一个字，则称集合 S **生成** G。

群 G 中关于子集 $S = \{s_1, \cdots, s_n\}$ 的**字问题**为，判断 S 中的两个字是否相等，也就是说，判断这两个字是否给出群 G 中相同的元素。群 G 中关于 S 的**共轭问题**就是，对于 S 中的两个字 x, y，判断是否存在 $g \in G$ 使得 $gxg^{-1} = y$。众所周知，通常来说字问题具有不可决定性，意味着没有算法可以解决它。共轭问题处在同样状况。即使对于特殊的群存在某个算法，例如 Coxeter 群（见[Garret 1997]），这种算法也可能是“不好的”，也就是说，它的运行时间与输入的长度成指数关系。

因此，通常来说字问题和共轭问题都是困难问题。但是在[Birman, Ko, Lee 1998]中，有一类被称为*辫群*的群非常有名，在其中字问题具有多项式时间解法，且似乎共轭问题还是没有解法。如果我们考虑这个条件，可以根据这类情况构造密码，其中要求授权解密者解决字问题，而要求非授权解密者解决共轭问题。

具有 n 个生成元的**辫群**是由集合 $S = \{s_1, \cdots, s_n\}$ 生成的群 G，且我们知道：

$$s_i s_{i+1} s_i = s_{i+1} s_i s_{i+1}$$

以及，对于 $|i - j| > 1$ 有：

$$s_i s_j = s_j s_i$$

这里不存在隐含的关系，即从某种意义上讲，任何其他正确的等式都是从这个关系导出的。也就是说，除了具有相邻下标的元素 s_i，其他元素两两之间是可以交换的，而且相邻两个元素具有上述有趣的关系。这唤起了对实际“辫子”的联想。

辫群为**阿廷群**的定义的一个特殊情况，其中所有参数 m_i 都为 3（如下所述）。一个阿廷群为由集合 $S=\{s_1,\cdots,s_n\}$ 生成的一个群 G，且我们知道：

$$\underbrace{s_i s_{i+1} s_i \cdots s_i s_{i+1} s_i}_{m_i\text{个元素}} = \underbrace{s_{i+1} s_i s_{i+1} \cdots s_{i+1} s_i s_{i+1}}_{m_i\text{个元素}}$$

且对于 $|i-j|>1$ 有：

$$s_i s_j = s_j s_i$$

而且，不存在“隐含”关系，即其他任何正确的等式都是从这个关系导出的。也就是说，除了具有相邻下标的元素 s_i，它们两两之间是可以交换的，而且相邻两个元素具有上述关系，如果 $m_i=3$ 就得到上面一个段落中的辫群。**Coxeter 群**的定义包含阿廷群的定义，此外还满足一个额外的关系，即对于所有的指数 i 满足：

$$s_i^{\,2} = e$$

备注 由于缺少初步的经验，所以这看起来确实有些神秘！而且，理解这类知识确实存在困难。实际上，如果进行适当安排的话，我们认为共轭问题在本质上和任何计算问题一样困难。

密钥交换。如果必须在不安全的信道上进行通信，密钥交换的思想就是为了建立一个共享的秘密。Alice 和 Bob 执行如下步骤。公共信息为：群 G，两个序列 $S_A=\{a_1,\cdots,a_m\}$ 和 $S_B=\{b_1,\cdots,b_n\}$ 为群 G 的元素。Alice 在 S_A 中选择一个秘密字 a，Bob 在 S_B 中选择一个秘密字 b（记住这意味着 a 可以用 S_A 中的元素及其逆来表示，b 也一样）。Alice 将下列序列发送给 Bob：

$$ab_1a^{-1}, ab_2a^{-1}, \cdots, ab_na^{-1}$$

而 Bob 则将下列序列发送给 Alice：

$$ba_1b^{-1}, ba_2b^{-1}, \cdots, ba_mb^{-1}$$

（这些序列必须稍微伪装使得操作能够正常进行），则 Alice 和 Bob 的公共密钥为下列表达式（称为 a 和 b 的换位子）：

$$\text{公共密钥} = aba^{-1}b^{-1}$$

我们可以通过下面的方法来计算它：

令：

$$a^{-1} = a_{i_1}^{\,e_{i_1}} \cdots a_{i_N}^{\,k_{i_N}}$$

为 Alice 的密钥 a 的逆 a^{-1} 关于其 a_i 的一个表达式，则**共轭运算** $x \to bxb^{-1}$ 的一个基本特性为：

$$\begin{aligned} ba^{-1}b^{-1} &= b(a_{i_1}^{\,e_{i_1}} \cdots a_{i_N}^{\,k_{i_N}})b^{-1} \\ &= (ba_{i_1}b^{-1})^{e_{i_1}} \cdots (ba_{i_N}b^{-1})^{k_{i_N}} \end{aligned}$$

现在，那些表达式 ba_jb^{-1} 就是 Bob 发送给 Alice 的值，因此她知道其值。也就是说，Alice 可以计算 $ba^{-1}b^{-1}$，而且因为她知道她自己的秘密 a，她可以计算：

$$a\cdot(ba^{-1}b^{-1}) = aba^{-1}b^{-1}$$

对称地，Bob 也可以计算 aba^{-1}，因为他知道他自己的秘密 b（及其逆 b^{-1}），他可以计算：

$$(aba^{-1})\cdot b^{-1} = aba^{-1}b^{-1}$$

因此，Alice 和 Bob 就具有了一个共享秘密。

备注 对此类密码最明显的攻击取决于是否能够解决共轭问题。例如，目前还不知道共轭问题在辫群中是否具有一个快速算法。

备注 上述描述并没有依赖于特殊的群 G。但是为了使得共享秘密 $aba^{-1}b^{-1}$ 更加明确，必须转化为某种“规范的”形式。对于辫群，这一点在[Birman, Ko, Lee 1998]中有效地实现了。

给定群 G 和 G 中生成元 $s_1,\cdots,s_n$ 的一个集合 S，对于元素 $w\in G$，选择 w 关于 s_i 的一个尽可能短的表达式：

$$w = s_{i_1}^{k_1}s_{i_2}^{k_2}\cdots s_{i_N}^{k_N}$$

也就是说，我们选择使下式为最小值的一个表达式：

$$|k_1|+|k_2|+\cdots+|k_N|$$

则 w 的**长度**为（关于 S）：

$$w\text{的长度} = l(w) = |k_1|+|k_2|+\cdots+|k_N|$$

这个定义在具有生成元 S 的任意群 G 中都有意义，但是只有在一些具有特殊生成元的群中才具有易处理的特性，例如辫群、阿廷群和 Coxeter 群。通常有：

$$l(xy)\leqslant l(x)+l(y)$$

如果 x 和 y 的最短字表达式的某些部分被删除了，则乘积 xy 的长度有可能比 x 和 y 的单个长度的和要小得多。

这样，我们非常粗略地描述对算术密码的 Hughes-Tannenbaum 长度攻击。利用上面关于对密码描述的概念，凭直觉我们应该计算：

$$l(a_i^{\pm 1}(at_ja^{-1})a_i^{\mp 1})$$

只要满足：

$$l(a_i^{\pm 1}(at_ja^{-1})a_i^{\mp 1}) < l(at_ja^{-1})$$

我们把 $a_i^{\mp 1}$ 当作 a 的一个具有非零概率的因子。针对这个问题的一个大致分析表明，这里的工作量将为长度和密钥数量的多项式。

备注 这个攻击是一个比密码本身更新的思想，并且还没有完成全部的研究工作。可能它还不能完全破解此类密码，但是惟一需要指出的是，参数的选择必须要明智。这是一个需要继续研究的课题。

10.8 量子密码

因为在本章我们不准备介绍量子机制，所以我们对量子密码和量子算法的描述将是非常简单的。

首先，关于量子效应，我们不打算关注传统计算机部件尺寸的降低，也不准备关注量子效应在生产晶体管和芯片上长达几十年的使用。我们将简要讨论一些新奇的应用以及对量子效应的开发。

从某种性能意义上讲，似乎实用的**量子信道**可以而且正在被用于具有绝对安全性的通信。大致地讲，这些都是为那些对于量子效应非常敏感的信道而设计的，例如由于观察而出

现的状态变化。特别地，一般的思想是，如果有任何人在信道上进行监听，它就会改变消息。这也是由于基本的量子机制，因为“监听”可能成为“观察”。也就是说，一定能够检测到监听。使用这样的信道作为基本机制来确保通信安全是**量子密码**的一部分。尽管存在很多实际的限制，但工作模型已经制定出来了。这种机制的缺陷是，如果存在一个并不介意被检测到的监听者，则通信就会中断，而且没有防御机制，只有检测机制。

量子密码有一个似乎不太实用的性质，即所谓的**量子心灵运输**（quantum teleportation），它意味着使用了 Einstein-Podolsky-Rosen 效应[Einstein, Podolsky, Rosen 1935]。这与 Bell 的理论[Bell 1993]相关。大致的思想为，如果两个质点具有处于混乱状态的量子状态，则它们在空间中是分离的，一个状态(也许是通过观察)的变化立刻会引起另一个状态的变化。也就是说,信息传输的速度比光速还快，任何干扰介质对其都没有影响。假如我们能用一种可信的方式生产这种东西，这将是一个量子心灵运输，将提供一个绝对安全的难以想像的通信。

另一种不同的情况是，制造一个真正的量子计算机的进程似乎确实在不断在加快。尽管普通晶体管和微芯片很好地利用量子效应已有许多年了，大部分的设计者仍希望那些电路以某种可与肉眼看到的物体相比较的方式运行，这实际上是为了避免量子行为的特殊性。到 20 世纪 80 年代一些人想利用量子效应而不是避免量子效应。早些时候 Richard Feynman 曾指出，在传统的计算机中不可能有效地模拟量子事件。

量子计算机假设的讨论大约到 1993 年还有些不太清楚，当时 Peter Shor 关于在量子计算机上[Shor 1996a]快速分解大数的算法确定地表明，如果并且当能够制造出一台实用的量子计算机，则大型计算中将发生质的变化。例如，大多数流行的公钥密码，如 RSA，将彻底被攻破。

虽然在 20 世纪早期，对量子计算机的热心者很乐观,但实际的进展却比想像的慢得多。最近的一些进展在[Knill,Laflamme,Martinez,Tseng2000]中有过介绍，那个研究小组早先称他们成功地实现了对 3 qubits 的处理，然后是 5 qubits,而现在是 7 qubits。在所有情况下他们都利用了核磁共振(NMR)。用他们的话说，“我们的实验程能够用作一个可信和有效的方法产生一个标准的伪随机状态，这是在液态 NMR 系统中实现传统量子算法的第一步。”

另一方面,即使我们很乐观地想像量子计算机很快就能制造出来，那么低级的传统计算技术也必须在量子计算机中重新确立或被取代。例如,在相同的自然（Nature）问题中[Pati,Braunstein 2000]，据说不可能删除任意量子状态的拷贝。而传统的计算机能删除信息,并且可用备份来撤消删除，但量子计算机不能删除量子信息。

在传统计算机上已经不存在的一个关键技术问题就是(内部)纠错和容错计算问题。在这一问题上,传统计算机是如此可先靠以致于几乎不需要建立内部冗余。相比而言，被量子计算机利用的量子效应则本质上需要开发一个全新的面向量子的纠错技术。虽然这个技术大致可建立在传统纠错的基础之上，但量子机制的特殊性则要求对其稍微做些改变。作为一个例子，请参见[Calderbank,Shor 1996]。

总之，人们的期望很高，但实际的进展并不是那么好。现在仍然不知道是否会存在量子计算机。如果量子计算机真的存在，也不清楚一般人能否用得起。更不清楚是否允许一般人拥有它们。

习题

10.8.01(**) 构建一个量子计算机能够将 6 分解因数为 2×3。

10.8.02(**) 构建一个量子计算机能够将 15 分解因数为 3×5。

10.9 美国出口限制

直到最近几年，密码产品还被美国政府归类为军需品，其出口需要联邦政府的正式批准。通常，强密码产品完全不可能出口，尽管强密码产品在全世界都可以得到的事实并不使人惊讶。很多年来，人们都推测这一限制后面的动机。很多人推断，美国相信如果它不对强密码产品进行出口，世界上的其他人就不可能得到强的加密算法。另外一些人认为，禁止出口将阻碍强加密算法在国内的推广使用，而联邦调查局（F.B.I）当然希望阻止强加密算法的使用。

到了 1998 年，这种严格的控制得到了一定程度的放松：用于鉴别目的的任意密钥长度的 RSA 软件产品可以出口，尽管必须证明此类产品不可能很容易被用于加密的目的。例如用于加密的 RSA 算法，其密钥长度被明确限制为 512 位。用于商业用途而具有更大密钥长度的加密产品的出口有时也是允许的：例如，用于商业交易的使用 768 位密钥的 Cybercash 被允许出口。

到 1999 年底，联邦立法至少在原则上改变了这一状况。从理论上讲，一旦得到联邦政府的批准，任何东西都可以出口，只要强密码产品不被出口到支持恐怖主义的国家。这就使得一些公司可以出口强密码产品，但是不清楚私人的状况。例如，不清楚是否一些强密码产品的“免费”代码是否可以被合法地放到 Web 站点上。还有一些另外的问题，例如将某些东西放在 Web 上是否被当作出口。

2000 年 5 月，欧盟非正式地宣布了一个暂时的协定而使世界震惊，即对欧盟成员以及其他一些国家，包括美国、加拿大和日本完全解除对密码软件的出口管制。这一做法与美国政府政策制定者的出口政策完全相反，也与英国和法国的政策相反。假设美国政府相关政策的宽松性将接近或者超过欧盟的政策，将使得美国生产商具有竞争力。欧盟达成的暂时协定计划在 2000 年 6 月 13 日得到正式批准并正式公布，同时还公布其他一些最近制定的政策，但是最终并没有公布。欧盟没有给出任何解释。难道又回到了过去的状况？

对于目前的大多数出口问题的状况而言，RSA 密码算法的 FAQ 是一个合理的参考，对于 RSA 有关的实际问题也一样，请访问如下 Web 页：

http://www.rsa.com/rsalabs/newfaq/

第11章 素 数

本章将详细讨论素数。除第一小节涉及到欧几里得对素数有无穷多个的证明之外，其余各节的讨论比本书的其他部分都要复杂，但是跳过这部分内容不会影响对本书其余内容的理解。

但是为了加深理解，读者最好还是仔细阅读本章内容，并参考有关资料以便了解更加详细的内容。

11.1 欧几里得定理

我们的经验可能已经表明，整数可以惟一分解为素数的乘积，但是对于存在无限多个素数的结论，从直觉上看人们好象还不是那么认同。欧几里得在 2000 年前的证明不但具有独创性，而且是一个间接证明（反证法）的好例子。

在此讨论中，我们认可整数可以惟一分解为素数乘积的结论。（这是我们后面结论即所有欧几里得环具有惟一的因式分解的一个特例。）

定理（欧几里得） 素数有无穷多个。

证明 用反证法。假设只有有限多个素数。设 $p_1,\cdots,p_n$ 是全部的素数，考虑数 $N=p_1\cdot p_2\cdot\cdots\cdot p_{n-1}\cdot p_n+1$。也就是说 N 是所有素数之乘积再加 1。因为 $N>1$，且 N 可以分解为素数的乘积，故一定存在素数 p 整除 N。然而 p 一定不在素数 $p_1,\cdots,p_n$ 之列，因为如果 p 是其中之一，则 p 整除 $N-(p_1\cdots p_n)=1$，这显然不成立。因此实际上，p 不在 p_i 之列已经与我们的假设——列出了所有的素数产生了矛盾。那也就是说，假设只有有限个素数不成立，因此素数应有无限多个。 ♣

注意，这里并没有给出进一步的结论，即哪些数是或者不是素数，也没说明到底存在多少个那样的素数。

11.2 素数定理

大约 1800 年前，勒让德已经根据列出的素数对素数的属性进行了猜测。高斯也考虑了这个问题，但是勒让德和高斯都没有能够精确地证明任何结论。直到一百年后，即 1896 年，Hadamard 和 la Vallée－Poussin 分别独立证明了下面的结论。

素数的标准计数函数为：

$$\pi(x)=\text{比 } x \text{ 小的素数的个数}$$

我们使用标准的符号：

$$f(x)\sim g(x)$$

即：

$$\lim_{x\to+\infty}\frac{f(x)}{g(x)}=1$$

定理 素数定理：当 $x \to +\infty$ 时，$\pi(x) \sim \dfrac{x}{\ln x}$。

此定理的证明很困难，我们不打算在这里给出详细的推导。此理论正如其表达的那样，并没有指出表达式 $(\pi(x)\ln x)/x$ 是如何收敛到 1，以及变化过程是否有任何起伏波动。下一步的工作就是解决逼近这个极限的问题，对这个问题的研究在该定理证明之后的头十年十分活跃，并一直延续到今天。

11.3 序列中的素数

在欧几里得证明存在无限多个素数的 1900 年之后，狄利克雷才给出了在某个算术级数中有关的素数结论。

一个算术级数是表达式 $an+b$ 的集合，其中 $a \neq 0$ 且 b 为固定的整数，$n = 0,1,2,3,4,\cdots$。例如：假设 $a=5, b=3$，我们可以得到下列算术级数：

$$3,8,13,18,23,28,33,38,\cdots$$

假设 $a=3, b=6$，我们可以得到下列算术级数：

$$3,9,15,21,27,33,39,\cdots$$

在第一个序列中已经有几个素数，如果我们继续下去，就会得到更多的素数。第二个序列中的第一个元素为 3，也是惟一的一个素数。实际上，我们可以看到，第二个序列中的每一个元素都可以被 3 整除，因此当然不是素数（从第二个开始）。因此第二个序列一定不会包含无限多个素数。因此，根据这个例子的经验，我们可以认为，不能指望在一个算术序列 $\{an+b : n=0,1,2,\cdots\}$ 中得到无穷多个素数，除非满足 $gcd(a,b)=1$。最乐观的结论就是满足下面这个简单且易理解的条件：

定理（狄利克雷） 令 $gcd(a,b)=1$，则在下列算术序列中存在无穷多个素数：

$$b, a+b, 2a+b, 3a+b, 4a+b, \cdots$$

（证明过程省略，因为这对我们来说太难了。）

注释 如果满足 $b=1$，则利用分圆多项式可得到一个可理解的证明，该多项式在后面将会提到。

几乎就在 1896 年素数定理被证明之后，人们将该思想与狄利克雷早期关于算术级数中的素数的思想结合起来，证明了一个相关的命题：

定理 令 $gcd(a,b)=1$，设 $\pi_{a,b}(x)$ 为算术序列“$b, a+b, 2a+b, 3a+b, 4a+b, \cdots$”中素数的个数，序列中的数小于上界 x。设 $\varphi(a)$ 为 a 的欧拉函数，则有：

$$\lim_{x\to\infty} \frac{\pi_{a,b}(x)}{x/\varphi(a)\ln x} = 1$$

（证明过程省略，因为这对我们来说太困难。）

在后面定理中出现的 $\varphi(a)$ 的因子，充分表明了这样一个事实：如果 b 和 a 具有公因子，则在任何一个此类算术序列中都不可能出现素数。因此对于给定的 a，只有 $\varphi(a)$ 个算术序列中会出现素数（如果序列以 a 的不同倍数作为开始，并不认为它们是不同的）。因此我们说，在素数 b 模 a 的同余类中素数是**等分布的**，这里的 b 与 a 互素。

11.4　车贝雪夫定理

1851 年，车贝雪夫在素数定理的证明上取得了突破。尽管他所证明的结论比所猜测的结论要弱，但是除了收集统计数据和进行罗列之外，这是第一个实质的进展。根据我们所掌握的知识，他的证明或多或少是可理解的，因此我们在此给出证明。

定理（车贝雪夫）　存在正常数 c 和 C 满足（对足够大 x）：

$$c\cdot\frac{x}{\ln x}\leqslant\pi(x)\leqslant C\cdot\frac{x}{\ln x}$$

证明　我们需要定义标准的辅助函数：

$$\theta(x)=\sum_{\text{素数p：}\ p<x}\ln p$$

$$\psi(x)=\sum_{\text{素数p：}\ k\in Z,p^k<x}\ln p$$

也就是说，$\theta(x)$ 是比 x 小的所有素数的自然对数的和，$\psi(x)$ 是 $\ln p$ 的和，其中素数幂 p^k 比 x 小。最简单的估计来自于 θ 和 ψ，因此我们最后会回过头来看该定理是如何描述素数计数函数 π 的。第一个必要的事情是，为了渐进估算的目的，θ 和 ψ 不能差别太大。此外，车贝雪夫还给出了两个更聪明的引理。

引理　$0\leqslant\psi(x)-\theta(x)\leqslant x^{1/2}(\ln x)^2$。

（证明留给读者：此不等式不存在任何精妙之处。）

引理（车贝雪夫）　$\theta(x)=O(x)$。

证明　设 $m=2^e$，e 为正整数，考虑二项式系数

$$N=\binom{m}{m/2}$$

因为

$$2^m=(1+1)^m=\sum_{0\leqslant k\leqslant m}\binom{m}{k}1^{m-k}1^k=\sum_{0\leqslant k\leqslant m}\binom{m}{k}$$

很明显，$\binom{m}{m/2}$ 是一个正整数，且小于 2^m。另一方面，根据表达式：

$$\binom{m}{m/2}=\frac{m!}{(m/2)!(m/2)!}$$

我们可以得到：每一个素数 p（$\frac{m}{2}<p\leqslant m$）整除 $\binom{m}{m/2}$。也就是说，

$$\prod_{(m/2)<p\leqslant m}p\leqslant\binom{m}{m/2}$$

自然对数函数为**单调递增**的，即对于 $x<y$，有 $\ln(x)<\ln(y)$。因此，对等式的两边同时取自然对数，可以得到：

$$\theta(m)-\theta(m/2)\leqslant m\ln 2$$

也就是

$$\theta(2^e)-\theta(2^{e-1})\leqslant m\ln 2=2^e\ln 2$$

因此，重复此步骤，我们可以得到：

$$\begin{aligned}\theta(2^e)&=(\theta(2^e)-\theta(2^{e-1}))+\theta(2^{e-1})\\&\leqslant 2^e\ln 2+(\theta(2^{e-1})-\theta(2^{e-2}))+\theta(2^{e-2})\\&\leqslant 2^e\ln 2+2^{e-1}\ln 2+\theta(2^{e-2})\\&\leqslant 2^e\ln 2+2^{e-1}\ln 2+2^{e-2}\ln 2+\theta(2^{e-3})\end{aligned}$$

继续重复此步骤，有

$$\begin{aligned}&\leqslant 2^e\ln 2+2^{e-1}\ln 2+2^{e-2}\ln 2+\cdots+2^1\ln 2+\ln 2\\&=2^{e+1}\ln 2-1\leqslant 2^{e+1}\ln 2\end{aligned}$$

因此，对于$2^{e-1}\leqslant x\leqslant 2^e$，有

$$\theta(x)\leqslant\theta(2^e)=2^{e+1}\ln 2=2\cdot 2^e\ln 2\leqslant 4\cdot x\ln 2=(4\ln 2)\cdot x$$

于是证明了此引理。 ♣

引理（车贝雪夫） 存在正常数c和C满足不等式

$$cx\leqslant\psi(x)\leqslant Cx$$

证明 考虑：

$$I=\int_0^1 x^n(1-x)^n dx$$

乘以$(1-x)^n$，并进行逐项积分，没有一项的分母比$2n+1$要大，因此如果我们乘以$1,2,3,\cdots,2n+1$的最小公倍数，结果就可以得到一个整数：

$$I\times lcm(1,2,3,\cdots,2n,2n+1)\in\mathbf{Z}$$

因此有：

$$1\leqslant I\times lcm(1,2,3,\cdots,2n,2n+1)$$

或者：

$$\frac{1}{I}\leqslant lcm(1,2,3,\cdots,2n,2n+1)$$

另一方面，在区间$[0,1]$中，$x(1-x)$的最大值为$1/4$，因此被积函数的最大值为$(1/4)^n$，且有：

$$I\leqslant(\frac{1}{4})^n$$

上式可以变换为：

$$4^n\leqslant\frac{1}{I}$$

因此，将这些不等式组合起来，我们可以得到：

$$4^n\leqslant lcm(1,2,3,\cdots,2n,2n+1)$$

两边取对数可得：

$$(\ln 4)\cdot n\leqslant\ln lcm(1,2,3,\cdots,2n+1)$$

现在我们感到很高兴，因为可以得到下面本质的观察结论（也就是车贝雪夫第一次介绍符号

ψ 的真正原因)。

引理　$\psi(n)=\ln lcm(1,2,3,\cdots,2n+1)$。

(证明留给读者：其证明并不难!)

最后，我们可以回到计数函数 $\pi(x)$，而不是辅助函数 θ 和 ψ。我们得到一个**黎曼-斯蒂尔切斯积分**：

$$\pi(x)=\int_{3/2}^{x}\frac{1}{\ln t}d\theta(t)$$

进行分部积分可以得到：

$$\pi(x)=\frac{\theta(x)}{\ln x}+\int_{3/2}^{x}\frac{\theta(x)}{t\ln^2 t}dt$$

根据等式 $\theta(x)=O(x)$，得到：

$$\pi(x)=\frac{\theta(x)}{\ln x}+\int_{3/2}^{x}\frac{O(x)}{t\ln^2 t}dt=\frac{\theta(x)}{\ln x}+O\left(\frac{x}{\ln^2 x}\right)$$

因为我们现在知道：

$$cx\leqslant\theta(x)\leqslant Cx$$

所以对于某些正的常数 c 和 C，有：

$$\frac{cx}{\ln x}\leqslant\theta(x)\leqslant\frac{Cx}{\ln x}$$

证毕。　♣

11.5　最佳渐进法

关于素数的渐进分布，已知的最佳断言要比素数定理的结论稍强，因为它给出了一个误差项。此结论起源于 I.Vinogradov 和 Korobov 的工作，但是由 A.Walfisz 和 H.-E.Rickert 最终完成。见[Walfisz 1963]。对此内容的英文版说明以及相关结论可以见文献[Karatsuba 1990]。

对数积分被定义为：

$$\mathrm{li}(x)=\int_{2}^{x}\frac{1}{\ln t}dt\qquad(\sim\frac{x}{\ln x})$$

关于素数分布（1997）的最佳渐进断言的证明为：*存在一个正常数 c 满足*:

$$\pi(x)=\mathrm{li}(x)+O\left(\frac{x}{e^{c(\ln x)^{3/5}(\ln\ln x)^{-1/5}}}\right)$$

为了简单一些，我们可以将此断言弱化为：

$$\pi(x)=\frac{x}{\ln x}+O\left(\frac{x}{\ln^2 x}\right)$$

因为 $\mathrm{li}(x)$ 是单调递增的，它有一个反函数。令 $\mathrm{li}^{-1}(x)$ 为其反函数（不是 $1/\mathrm{li}(x)$），则第 n 个素数 p_n 可以估算为：

$$p_n=\mathrm{li}^{-1}(n)+O\left(\frac{n}{e^{(\ln n)^{3/5}(\ln\ln n)^{-1/5}}}\right)(\sim n\ln n)$$

11.6 黎曼假设

尽管素数定理在1896年以前没有得到证明，但是早在1858年，B.Riemann已经看出了素数分布的误差项之间的联系，以及一个特殊函数的精妙特性，即为**ζ函数**，定义如下。

对于实数部分大于1的复数s，序列：

$$\zeta(s)=\sum_{n\geq 1}\frac{1}{n^s}$$

是绝对收敛的，且是s的一个函数。这就是**ζ函数**，通常称作**黎曼函数**，因为G.Riemann（约1858年）是第一个注意到$\zeta(s)$的分析特性直接与素数分布的性质有关的人。其他人（例如，L.Euler）也已经看到了它们之间的一般联系。欧拉也已经观察到了**欧拉乘积扩展**：对于实数部分大于1的复数s，有

$$\zeta(s)=\prod_{\text{素数}p}\frac{1}{1-\frac{1}{p^s}}$$

给出此函数就意味着当s的实数部分小于或者等于1时，会出现问题。但是，这个问题已在140多年前被解决了，而且不是由欧拉解决的。

对于在范围$\frac{1}{2}<r<1$内的实数r，令PNT_r为如下命题：

$$\text{对于所有}\varepsilon>0\text{，有}\pi(x)=\frac{x}{\ln x}+O(x^{r+\varepsilon})$$

目前还没有证明这个结论是正确的，认识这一点很重要：此结论中的**误差项**渐进地小于已经被证明成立的结论中的任意误差项。

另一方面，同样对于在范围$\frac{1}{2}<r<1$内的实数r，令RH_r如下命题：

$$\text{当}s\text{的实数部分大于}r\text{时，}\zeta(s)\neq 0$$

迄今没有人能够证明这个命题对于任意$r<1$成立。同时，我们知道存在无穷多个复数ρ，若实数部分为$\frac{1}{2}$，$\zeta(\rho)=0$成立。

定理（由黎曼提出） 对于每一个$\frac{1}{2}<r<1$，命题PNT_r**等价于**RH_r。

特别地，**黎曼假设**为：对于实数部分大于1/2的复数s，有$\zeta(s)\neq 0$。

因此，如果已知黎曼假设是正确的，则在素数渐进分布的描述中，将可能得到最有可能的误差项。但在此方向上并无任何本质的了解，尽管积累的数字都表明黎曼假设是正确的。如果黎曼假设是正确的，实际上（如同H.vonKoch所证明的那样），

$$\pi(x)=\text{li}(x)+O(\sqrt{x}\ln x)$$

$$\text{第}n\text{个素数}=\text{li}^{-1}(n)+O(\sqrt{n}(\ln n)^{5/2})$$

后来出现了**扩展黎曼假设**，这是一个与比ζ函数更广的函数类的零值相类似的论断。并且**推广的黎曼假设**也是关于一个更广的函数类的零值的论断。如果所有这些假设都是正确的，则它们给出关于素数分布以及素数广义性的最可能错误估计。不幸的是，除了支持这些假设的一些数字以外，人们对它们尚无任何本质的了解。

第12章 mod p的根

早在350年以前，皮埃儿·德·费马就对素数、素数分解以及数论的相关方面做了许多细致入微的观察（这并不涉及数学和自然科学的其他方面）。大约在300年前，欧拉系统地延续了费马的工作。他所做的大多数事情都成为“现代”数学思想的原型，与此同时也留下了许多与现代数论及其应用密切相关的问题。

本章应用了费马的几个思想。所谓的费马小定理是其后几个结论的原型，并且紧接着应用在因式分解技巧以及模素数根的公式中。

实际上，在本书的后续部分会不断使用到许多相当简便的计算在 $\mathbf{Z}/p$ 域上的指数和根的方法。

12.1 费马小定理

这个结果已经有超过350年的历史。这个结果本身就是初等数论中一个基本结论，并且是第一个概率的素性检验的起源。只要用很少的条件就可以证明费马小定理，我们现在开始证明这个定理。

定理 p 是一个素数。对任意整数 x 有

$$x^p \equiv x \bmod p$$

证明 我们将首先证明，当 $1 \leqslant i \leqslant p-1$ 时，素数 p 整除二项式系数

$$\binom{p}{i}$$

请记住在“极端”的情况 $i=0$ 和 $i=p$ 时不可能也满足这个性质，由于

$$\binom{p}{0}=1 \qquad \binom{p}{p}=1$$

实际上，根据其定义，

$$\binom{p}{i}=\frac{p!}{i!(p-i)!}$$

p 显然整除分子。由于 $0<i<p$，所以素数 p 不整除所有在分母中阶乘的因子。根据素数因子分解的惟一性，这就是说不能整除分母。

根据二项式定理，

$$(x+y)^p=\sum_{0\leqslant i\leqslant p}\binom{p}{i}x^i y^{p-i}$$

特别的，由于左边的系数是整数，所以右边也必须是整数。因此，所有二项式系数都是整数。(我们不使用 p 是素数这个条件来得出结论。)

因此，当 $0<i<p$ 时，二项式系数是整数并且其分式形式中的分子可以被 p 整除，而分

母不能被 p 整除。所以，在分式化简完成后，分子中肯定存在因子 p。这就证明了有关二项式系数的所需条件。

现在，我们通过归纳 x（对正整数 x）来证明费马小定理。首先，显然 $1^p \equiv 1 \bmod p$。根据归纳法的步骤，假设对特定的整数 x，存在

$$x^p \equiv x \bmod p$$

然后有

$$(x+1)^p = \sum_{0 \le i \le p} \binom{p}{i} x^i 1^{p-i} = x^p + \sum_{0 < i < p} \binom{p}{i} x^i + 1$$

等式右边的中间部分的所有系数整除 p，因此

$$(x+1)^p \equiv x^p + 0 + 1 \equiv x + 1 \bmod p$$

由于我们的归纳假设 $x^p \equiv x \bmod p$。这就证明了定理中 x 是正整数的情况。

为了证明定理中 $x<0$ 的情况，我们应用 $-x$ 是正整数这个条件。如果 $p=2$，我们只需要直接分别证明 $x \equiv 0 \bmod 2$ 和 $x \equiv 1 \bmod 2$ 这两种情况。如果 $p>2$，则满足这个条件的 p 是奇数。因此，应用正整数的结果，

$$x^p = -(-x)^p \equiv -(-x) \bmod p = x \bmod p$$

至此定理得证。 ♣

习题

12.1.01 快速计算 $2^{1000}\%17$。

12.1.02 快速计算 $3^{1000}\%17$。

12.1.03 快速计算 $2^{1000}\%23$。

12.1.04 快速计算 $3^{1000}\%23$。

12.1.05 快速计算 $2^{1\,000\,000}\%17$。

12.1.06 快速计算 $3^{1\,000\,000}\%17$。

12.1.07 如果 p 是素数且 $gcd(x,p)=1$，那么 x 模 p 的一个乘法逆元素是 $x^{p-2}\%p$。

12.1.08(*) 试比较用欧几里得算法和前一个练习所给出的公式求乘法逆元素，哪一种方法更简便？

12.2 特殊的因式分解表达式

通过费马小定理，我们可以利用它的结论并且提高某些特殊的因式分解的速度。首先，我们证明一个引理，结论非常有用：

引理 如果 $b>1$，对任意两个正整数 m,n，有

$$gcd\ (b^m - 1, b^n - 1) = b^{gcd\ (m,n)} - 1$$

备注 在初等代数中对正整数 N，有恒等式

$$x^N - 1 = (x-1)(x^{N-1} + x^{N-2} + \cdots + x^2 + x + 1)$$

对于 n 的正因子 d，如果 $x = b^d$ 并且 $N = n/d$，我们可以得到

$$b^n - 1 = (b^d)^N - 1 = (b^d - 1)((b^d)^{N-1} + (b^d)^{N-2} + \cdots + (b^d)^2 + (b^d) + 1)$$

因此，如果 $d \mid n$，很容易得到 $b^d - 1$ 整除 $b^n - 1$。

证明：首先，需要注意的是，如果 $m=n$，可以断言这个命题是真的。下一步就是对更大的 m,n 使用归纳法。我们可以假设 $m \leqslant n$（如果需要的话，可以调换 m,n 的位置）。如果 $m=n=1$，断言 $gcd(b-1,b-1)=b-1$ 是真的。根据归纳法，由于 $m=n$ 的情况已经证明，我们可以假设 $m<n$，则有

$$(b^n-1)-b^{n-m}(b^m-1)=b^{n-m}-1$$

我们断言

$$gcd\ (b^m-1,b^n-1)=gcd\ (b^m-1,b^{n-m}-1)$$

一方面，如果 $d \mid b^n-1$ 并且 $d \mid b^m-1$，则 $d \mid (b^n-1)-b^{n-m}(b^m-1)$，那么 $d \mid b^{n-m}-1$。因此，b^n-1 和 b^m-1 的任意公因子 d 同时也是的 $b^{n-m}-1$ 因子。另一方面，重新整理表达式可得

$$b^n-1=b^{n-m}(b^m-1)+b^{n-m}-1$$

任何 b^m-1 和 $b^{n-m}-1$ 的公因子整除等式的右边，也就整除 b^n-1。这就证明了前一个断言。

因此，调用归纳假设，我们有

$$gcd\ (b^m-1,b^n-1)=gcd\ (b^m-1,b^{n-m}-1)=b^{gcd\,(m,n-m)}-1$$

所以我们应该证明 $gcd(m,n)=gcd(n-m,m)$：这和证明上一个断言的思路是一致的：一方面，如果 d 是 m 和 n 的公因子，那么 $d \mid n-m$；另一方面，应用 $n=m+(n-m)$，如果 d 是 m 和 $n-m$ 的公因子，那么也 $d \mid n$。 ♣

推论 对给定的正整数 b，n 是一个正整数，如果素数 p 整除 b^n-1，那么对 n 的因子 d，$d<n$，有 $p \mid b^d-1$，或者 $p \equiv 1 \bmod n$。

证明 如果 p 整除 b^n-1，根据费马小定理有 $b^{p-1} \equiv 1 \bmod p$，所以 p 整除 $b^{p-1}-1$。因此，根据引理，p 整除 $b^{gcd\,(n,p-1)}-1$。如果 $d=gcd(n,p-1)<n$，那么 $d<n$ 一定是 n 的正因子，并且满足 $p \mid b^d-1$。如果 $gcd(n,p-1)=n$，那么 $n \mid p-1$，也就是说 $p \equiv 1 \bmod n$。 ♣

备注 后一个推论说明诸如 b^n-1 这种数的因子在形式上是相当严格的。而且，对奇素数 p 和奇数 n，由于 $gcd(n,2)=1$，如果 $n \mid p-1$，那么从 $2 \mid p-1$ 我们可以得出 $2n \mid p-1$，所以 $p \equiv 1 \bmod 2n$。

12.3 梅森数

像上面那样，对形如 b^n-1 的数的可能素因子存在某些限制，由此极大地缩短了分解所需的时间，我们来分解梅森数 2^n-1。

例子 分解 $127=2^7-1$：由于 7 是素数，所以根据推论可得，这个数可能的素因子 p 必须满足 $p \equiv 1 \bmod 14$。另一方面，$\sqrt{127}<12$，所以我们只需要尝试所有小于 12 的素数。但不存在这样的满足模 14 同余 1 的数，所以 2^7-1 肯定是个素数。

备注 尽管通过手工的方式也可以很容易的检验出 127 的素数性，但我们通过“抽象的简单的思路”(就是说无需大量计算)能得到的相同结果即是个很好的例证。

例子 分解 $255=2^8-1$：指数是合数会产生许多因子：从 $2^2-1=3$ 我们得到 3，从 $2^4-1=15=3\cdot 5$ 我们得到 5。做除法，我们得到

$$255/(3\cdot 5)=17$$

是一个素数，所以 $2^8-1=3\cdot 5\cdot 17$。

例子 分解 $511=2^9-1$：合数指数产生一个因子 $2^3-1=7$，$511/7=73$ 是个素数。所以，$2^9-1=7\cdot 73$。

例子 分解 $1023=2^{10}-1$：首先，由于指数是个合数，那么这个梅森数肯定是个合数。我们注意到10的(正)因子中比10小的数是2和5，所以 $2^{10}-1$ 有因子 $3=2^2-1$ 和 $31=2^5-1$。其次，推论告诉我们其他整除 $2^{10}-1$ 的素数必须模10同余1。先试验11：则有

$$1023/11=93=3\cdot 31$$

因此

$$1023=3\cdot 11\cdot 31$$

当然，由于 $2^{10}-1$ 有小因子3，$1023/3=341$ 可能已经足够小了，因此可以通过手工方式继续进行因数分解。特别是已经知道了因子 $31=2^5-1$，$341/31=11$，结果就已经出来了。

例子 分解 $2047=2^{11}-1$：由于11是个素数，所以没有任何“容易得出的”因数。如果这个数被证明是素数，则它是**梅森素数**。上面的推论保证任何能够整除2047的素数 p 必须满足 $p\equiv 1 \bmod 11$。由于 p 肯定是个奇数，如上面提到的，我们实际上可以断定这样的 p 满足 $p\equiv 1 \bmod 22$。所以我们试着用 $23=22+1$ 去除2047，得到结果 $2047/23=89$（由于 $89<100$，仅需要试除2,3,5,7，说明89是素数即可）。由此可得 $2047=23\cdot 89$。

例子 分解 $4095=2^{12}-1$：由于指数是合数，所以可以很容易的从 $d<12$ 并且 d 整除12求出形如 2^d-1 的因子。也就是说，我们首先找到因子 $2^2-1=3$，$2^3-1=7$，$2^4-1=15=3\cdot 5$，和 $2^6-1=63=3^2\cdot 7$。因此，4095可以被 $3^2\cdot 5\cdot 7$ 整除，我们得到

$$4095/(3^2\cdot 5\cdot 7)=13$$

所以完整的因式分解为 $4095=3^2\cdot 5\cdot 7\cdot 13$

例子 分解 $8191=2^{13}-1$。由于指数13是素数，所以没有明显的因子。如果这个数被证明是素数，则它是**梅森素数**。由于 $\sqrt{8191}\approx 90.5$，所以我们只需要试除小于90的素数。上面的推论告诉我们只需考虑素数 $p\equiv 1 \bmod 26$。首先，$26+1=27$ 不是素数，所以我们不必考虑。其次，$2\cdot 26+1=53$ 是素数，但 $8191\%53=29$。最后，$3\cdot 26+1=79$ 是素数，但 $8191\%79=54$。因此8191是素数。

例子 分解 $16383=2^{14}-1$。我们首先得到 $2^2-1=3$ 和 $2^7-1=127$。我们已经知道127是素数。所以我们可以将素因子3和127排除，剩下

$$16383/(3\cdot 127)=43$$

这是一个素数。

例子 $32767=2^{15}-1$：从 $2^3-1=7$ 和 $2^5-1=31$ 中我们得到素因子7和31。将之去除，得到

$$32767/(7\cdot 31)=151$$

我们可以手工尝试，或者应用推论将素因子 p 的范围缩小在 $p\equiv 1 \bmod 30$ 而且还要小于 $\sqrt{151}<13$。由于没有满足条件的素数，我们可以根据性质得到151是素数。因此，素因子分解表达式为 $2^{15}-1=7\cdot 31\cdot 151$。

例子 $65535=2^{16}-1$：根据 $2^2-1=3$，$2^4-1=15=3\cdot 5$ 以及 $2^8-1=255=3\cdot 5\cdot 17$，我们得到素因子3、5和17，整除后得到

$$65535/(3\cdot 5\cdot 17)=257$$

现在我们可以应用推论通过 $p\equiv 1 \bmod 16$ 来缩小可能存在的素因子 p 的范围。与此同时，$\sqrt{257}<17$。没有符合条件的素数，所以257是素数，素因式分解表达式为

$$2^{16}-1=3\cdot 5\cdot 17\cdot 257$$

例子 分解 $131071 = 2^{17} - 1$：由于 17 是素数，所以我们只能通过 $p \equiv 1 \bmod 34$ 以及 $p \leqslant \sqrt{131071} < 362.038$ 来寻找素因子 p。首先，$34 + 1 = 35$ 不是素数；其次，$2 \cdot 34 + 1 = 69$ 能被 3 整除，所以不是素数；接下来，$3 \cdot 34 + 1 = 103$ 是素数，但 $131071 \% 103 = 55$；$4 \cdot 34 + 1 = 137$ 是素数，但 $131071 \% 137 = 99$；$5 \cdot 34 + 1 = 171$ 能被 3 整除；$6 \cdot 34 + 1 = 205$ 能被 5 整除；$7 \cdot 34 + 1 = 239$ 是素数（尝试所有小于等于 $\sqrt{239} < 16$ 的素除数 2,3,5,7,11,13），但 $131071 \% 239 = 99$。$8 \cdot 34 + 1 = 273$ 能被 3 整除。$9 \cdot 34 + 1 = 307$ 是素数（尝试所有小于等于 $\sqrt{307} < 18$ 的素除数），但 $131071 \% 307 = 289$。$19 \cdot 34 + 1 = 341$ 能被 11 整除。由于尝试了所有小于 362 的素数，所以 $131071 = 2^{17} - 1$ 肯定是个素数。

例子 证明 $524287 = 2^{19} - 1$ 是素数。如果要用最原始的方法进行验证，我们必须尝试所有小于等于 $\sqrt{524287} < 725$ 的素数因子，这样做大概需要进行 $725/2 \approx 362$ 次除法运算才能验证其素数性。但如果我们应用引理并且将素数 p 的范围缩小在 $p \equiv 1 \bmod 38$，则只需要进行 $725/38 \approx 19$ 次除法运算。

例子 分解 $8388607 = 2^{23} - 1$：根据推论，我们只需要验证素数 p，$p \equiv 1 \bmod 46$。而且幸运的是，47 整除这个数，得到 $8388607/47 = 178481$。如果要证明这是个素数，我们必须验证所有小于等于 $\sqrt{178481} < 422.5$ 的素数。实际只需要验证那些特殊的素数，大概需要进行 $422/46 \approx 9$ 次除法运算，而不是纯运算方法的 $422/2 \approx 211$ 次除法。

例子 分解 $536870911 = 2^{29} - 1$：任何可能的素因子 p 满足 $p \equiv 1 \bmod 58$，并且如果这个数不是素数，那它必定有小于等于 $\sqrt{536870911} \approx 23170$ 的因子。验证 59,117（不是素数），175（不是素数），233,…,幸运的看到 233 整除 536870911。得到：

$$536870911/233 = 2304167$$

后者不能被 233 整除。我们知道如果 2304167 不是素数，那么它存在小于等于 $\sqrt{2304617} < 1518$ 的素因子。在经过 14 次尝试之后，我们发现素数 $19 \cdot 58 + 1 = 1103$ 整除 2304167。得到 $2304167/1103 = 2089$。如果 2089 不是素数，那么存在小于等于 $\sqrt{2089} < 46$ 的素因子，但同时满足 mod 58 同余 1。不存在这样的素因子，所以 2089 是素数。因此，

$$536870911 = 233 \cdot 1103 \cdot 2089$$

12.4 更多的例子

我们继续给出费马求解形如 $b^n - 1$ 这种数的因子的更多例子。

对每一个 $3^n - 1$（$n > 1$）都明显有一个因子 $3 - 1$，所以它不是素数。但由于我们希望得到整个素因子分解表达式，或者至少想知道 $(3^n - 1)/(3 - 1)$ 是否是素数，所以这是一个相当弱的结论。费马的技巧对实现上述意图很有帮助，而且对寻找梅森数也是有益的。

已有的结论告诉我们如果素数 p 整除 $b^n - 1$，那么对 $d \mid n$，其中 $d < n$，有 $p \mid (b^d - 1)$ 或者 $p \equiv 1 \bmod n$。并且当 n 是奇数时，$p \equiv 1 \bmod 2n$。因此，不太严格地讲，通过考查 n 或 $2n$ 的因子，需要验证整除 $b^n - 1$ 的素数 p 的个数减少了。

首先，$3^2 - 1 = 8 = 2^3$。

下一步，$(3^3 - 1)/2 = 26/2 = 13$。费马的结论指出一个素数整除 $3^3 - 1$ 而不整除 $3^1 - 1$ 应该同余 $1 \bmod 2 \cdot 3 = 6$，比如 13。

下一步，$3^4 - 1 = (3^2 + 1)(3^2 - 1)$ 我们已经分解了 $3^2 - 1$。因子 $3^2 + 1$ 还是能被 2 整除，则 $(3^2 + 1)/2 = 5$，这是素数且同余 $1 \bmod 4$。

下一步，$3^5-1=242$。除去因子$3-1=2$剩下121。根据费马的结论，任何能够整除这个数的素数必定模$2\cdot5=10$同余1。尝试$10+1=11$，我们看到实际上$121=11^2$。

下一步，$3^6-1=(3^3-1)(3^3+1)=(3^3-1)(3+1)(3^2-3+1)$。除了$3^2-3+1=7$之外，我们已经遇到过其他因子。而且，这个素数模6同余1。

下一步，$3^7-1=2186$。除去因子$3-1=2$剩下1093，根据费马的结论，我们知道它的任何素因子必须是同余$1 \bmod 2\cdot7=14$。由于$14+1=15$不是素数，所以我们尝试除1093的第一个素数是$2\cdot14+1=29$，而且我们看到$1093\%29=20$。因为$\sqrt{1093}\approx33.06<34$，所以我们预先就知道不需要在大于33的数中寻找1093的因子。但29是惟一小于34并且模14同余1的素数，所以我们的出结论：1093是素数。

下一步，$3^8-1=(3^4-1)(3^4+1)$。用因子$3-1=2$除3^4+1，得到$(3^4+1)/2=41$。费马的结论保证任何不整除$3^4-1=80=2^4\cdot5$而整除3^8-1将模8同余1，由于小于等于$\sqrt{41}<7$的数中都不满足模8同余1，所以41是素数（是的，这点我们早就知道）。

下一步，$(3^9-1)(3^3-1)=3^6+3^3+1=757$。根据费马的结论，757只可能被模$2\cdot9=18$同余1的素数整除。而且，如果757不是素数，那么其素因子小于等于$\sqrt{757}<28$。这个范围中的惟一素数是$18+1=19$，但$757\%19=16$，所以757是素数。

下一步，$(3^{10}-1)/(3^5-1)=244$。除去因子2^2，剩下61。由于没有能整除$3^2-1=8$或$(3^5-1)/2=11^2$的素因子，所以其任何素因子必须满足模10同余1，这样的素数不满足小于等于$\sqrt{61}<8$的条件，所以61是素数。（是的，这也是我们知道的。）

下一步，$(3^{11}-1)/2=88\ 573$任何整除它的素数必须满足模22同余1。尝试用诸如23这样的数去整除3851。首先，$3851\%23=10$，所以23不整除3851。如果3851不是素数，那它就有小于等于$\sqrt{3851}\approx62.06<63$的素因子，下一个候选者$23+22=45$不是素数。再下一个$45+22=67$是素数，但这个数太大了。所以，3851是素数。

即使使用费马的方法，要证明$(3^{13}-1)/2=797{,}161$是素数仍需要进行35次除法尝试。通过“手工”的方法勉强可以接受，不过比起原来验证所需要的400次除法尝试相比，这已经改进很多了。

省略一些步骤，我们来看$3^{15}-1$。商$(3^{15}-1)/(3^5-1)=59\ 293$的素因子可以整除$3^3-1=2\cdot13$或者满足模$2\cdot15=30$同余1。检验13，可以得到$59\ 293/13=4561$。（并且13不能整除4561。）这个数仅有的素因子肯定模30同余1。而且，如果4561不是素数，它必有一个小于等于$\sqrt{4561}\approx67.54<68$的因子。检验31，我们得到$4561\%31=4$。检验61，我们得到$4561\%61=47$。因此，4561是素数。

习题

12.4.01(*) 证明如果a,b互素，费马的结论同样适用于表达式a^n-b^n。也就是要证明，如果一个素数p整除a^n-b^n，那么对n的因子$d<n$，有$p\mid a^d-b^d$，或者$p=1 \bmod n$。

12.4.02 应用前面练习题的结论，对$1\leqslant n\leqslant16$，分解3^n-2^n。

12.4.03 应用第一个练习题的结论，对$1\leqslant n\leqslant16$，分解4^n-3^n。

12.4.04 简单（例如，通过简短的手工计算）证明$15^3-14^3=631$是素数。

12.4.05 简单（例如，通过简短的手工计算）证明$11^5-10^5=61\ 051$是素数。

12.4.06 简单（例如，通过简短的手工计算）证明$242\ 461=4^9-3^9$的因子6553是素数。

12.5 指数算法

进行指数运算最直接的方法，比如计算 x^n，就是先计算 x^2，然后计算 $x^3 = x \cdot x^2$，再计算 $x^4 = x \cdot x^3$，…，$x^n = x \cdot x^{n-1}$，这样做效率很低。在这里我们给出一个非常简单但有很大改进的方法，这个方法已经有至少3000年的历史。此改进跟模 m 的指数运算关系密切。

这个方法的思想是在计算 x^e 时，将 e 表示为一个二元整数

$$e = e_0 + e_1 \cdot 2^1 + e_2 \cdot 2^2 + \cdots + e_n \cdot 2^n$$

其中每个 e_i 等于0或1，并且通过计算 x 方幂的**平方**：

$$x^2 = x \cdot x$$
$$x^4 = (x^2)^2$$
$$x^8 = (x^4)^2$$
$$x^{2^4} = (x^8)^2$$
$$x^{2^5} = (x^{2^4})^2$$
$$\cdots$$

那么

$$x^e = x^{e_0}(x^2)^{e_1}(x^4)^{e_2}(x^8)^{e_3}(x^{2^4})^{e_4}\cdots(x^{2^n})^{e_n}$$

而且，e_i 的值为0或1，所以实际上这种记法显得不够简洁，可进行化简：如果 $e_k = 0$，忽略 x^{2^k} 项；如果 $e_k = 1$，包含 x^{2^k} 项。

下面是一个相当好的实现方法。为了计算 x^e，我们将利用三元组 (X, E, Y)，其初始值为 $(X, E, Y) = (x, e, 1)$。算法步骤如下：

- 如果 E 是奇数，那么用 $X \times Y$ 代替 Y，用 $E-1$ 代替 E；
- 如果 E 是偶数，那么用 $X \times X$ 代替 X，用 $E/2$ 代替 E；
- 当 $E = 0$ 时，Y 的值为 x^e。

备注 这个算法最多需要 $2\log_2 E$ 步（尽管如此，这个数字增长得还是相当快）。

对于我们的目标，当结合模 m 来进行约简时，这个很好的快速指数算法就引起了我们的兴趣：重写的算法为计算 $x^e \% m$，我们利用三元组 (X, E, Y)，其初始值为 $(X, E, Y) = (x, e, 1)$。算法步骤如下：

- 如果 E 是奇数，那么用 $X \times Y \% m$ 代替 Y，用 $E-1$ 代替 E；
- 如果 E 是偶数，那么用 $X \times X \% m$ 代替 X，用 $E/2$ 代替 E；

当 $E = 0$ 时，Y 的值为 $x^e \% m$。

而且，这个算法最多需要 $2\log_2 E$ 步。当需要模 m 的指数运算时，运算次数总是小于 m^2 的。因此，比如计算

$$2^{1000} \% 1\ 000\ 001$$

需要进行大概 $2\log_2 1000 \approx 2 \cdot 10 = 20$ 次六位数的乘法运算。通常，我们有

命题 用上述算法计算 $x^e \% m$ 共使用 $O(\log e \log^2 n)$ 次位运算。

例如，如果我们直接计算 $2^{1000} \bmod 89$。根据上面的说明，我们有

X	E	结果	
2	1000	1	初始状态
4	500	1	E 是偶数：计算 X 的平方模 89

16	250	1	E 是偶数：计算 X 的平方模 89
78	125	1	E 是偶数：计算 X 的平方模 89
78	124	78	E 是奇数：结果乘以 X 模 89
32	62	78	E 是偶数：计算 X 的平方模 89
45	31	78	E 是偶数：计算 X 的平方模 89
45	30	39	E 是奇数：结果乘以 X 模 89
67	15	39	E 是偶数：计算 X 的平方模 89
67	14	32	E 是奇数：结果乘以 X 模 89
39	7	32	E 是偶数：计算 X 的平方模 89
39	6	2	E 是奇数：结果乘以 X 模 89
8	3	2	E 是偶数：计算 X 的平方模 89
8	2	16	E 是奇数：结果乘以 X 模 89
64	1	16	E 是偶数：计算 X 的平方模 89
64	0	45	E 是奇数：结果乘以 X 模 89

我们得出结论

$$2^{1000}\%89 = 45$$

习题

12.5.01 用快速指数算法计算 $2^{56}\%1001$。

12.5.02 用快速指数算法计算 $3^{59}\%1001$。

12.5.03 用快速指数算法计算 $2^{62}\%1003$。

12.5.04 用快速指数算法计算 $3^{70}\%1003$。

12.5.05 通过计算 $2^{57}\%59$ 找出 $2 \bmod 59$ 的乘法逆元素。解释为什么可以这么做。

12.5.06 通过计算 $7^{57}\%59$ 找出 $7 \bmod 59$ 的乘法逆元素。解释为什么可以这么做。

12.5.07 通过计算 $23^{519}\%521$ 找出 $23 \bmod 521$ 的乘法逆元素。解释为什么可以这么做。

12.5.08 通过计算 $29^{519}\%521$ 找出 $29 \bmod 521$ 的乘法逆元素。解释为什么可以这么做。

12.5.09 运用快速指数算法计算矩阵 $\begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}$ 的第17次幂。

12.5.10 估计通过快速指数算法计算 2^{100} 所必需的位运算的数量。

12.5.11 定义斐波那契数，初始数为 $F_0 = 0$， $F_1 = 1$

关系式为

$$F_i = F_{i-1} + F_{i-2}, \quad i > 1$$

如果令

$$F = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

通过归纳法证明

$$F^n = \begin{pmatrix} F_{n-1} & F_n \\ F_n & F_{n+1} \end{pmatrix}$$

12.5.12 利用上一个练习题的结果，简单计算 $F_{128}\%10$（例如，用手工计算）。

12.6 mod p 的二次根

如果一个素数 p 满足 $p=3 \bmod 4$，那么对一个模素数 p 的平方数求其二次根，我们可以给出一个公式。由于我们在指数运算方面已有了一个好的算法，所以这个公式应该被看做一个求解二次根的好方法。需要注意的是，只有在素数模 p 是同余 3 模 4 时才适用，而且给定的数必须是 mod p 的一个平方数。（否则，公式得出的结果没有任何用处。）

定理 p 是一个满足 $p=3 \bmod 4$ 的素数。那么对模 p 的平方数 y，

$$x = y^{(p+1)/4} \bmod p$$

是 y 模 p 的一个二次根。也就是说，$x^2 = y \bmod p$。这被称为 y 的**主二次根**。

备注 需要注意的是，如果 y 不是模 p 的平方数，公式所求出的值并不是 y 模 p 的二次根。而且，对 $p=1 \bmod 4$ 没有类似的求二次根的简单公式可用。

证明 首先注意除非 $p=3 \bmod 4$，否则表达式 $(p+1)/4$ 不是整数。假设 $y=x^2$。我们来检验如果 $z = y^{(p+1)/4}$，那么有性质 $z^2 = y \bmod p$。（注意我们并不能断言 $z=x$）那么

$$(y^{(p+1)/4})^2 = y^{(p+1)/2} = (x^2)^{(p+1)/2} = x^{p+1} = x^p \cdot x^1 = x \cdot x = y$$

根据费马小定理我们知道 $x^p = x \bmod p$。 ♣

备注 区别于这个定理，这个**主二次根**还可以用另一种形式表达：素数模 $p=3 \bmod 4$，如果 y 是 mod p 的一个非零平方数，即 $x^2 = y \bmod p$，那么 y 一定有**两个**二次根 $\pm x \bmod p$，其中一个 mod p 的平方数，另一个不是 mod p 的平方数。（这一点不能仅从其自身的 ± 号判断。）本身也是平方数的二次根是主二次根。我们可以给定一个自身是平方数的二次根以检验定理中的公式：假设 $x^2 = y \bmod p$ 和 $p = 4k-1$。那么

$$y^{(p+1)/4} = (x^2)^{(4k-1+1)/4} = x^{2k} = (x^k)^2 \bmod p$$

也就是说，定理中公式所求出的主二次根是 x^k 模 p 的平方。

习题

12.6.01 求 2 模 19 的主二次根。
12.6.02 求 6 模 19 的主二次根。
12.6.03 求 2 模 71 的主二次根。
12.6.04 求 50 模 71 的主二次根。
12.6.05 求 2 模 103 的主二次根。
12.6.06 求 2 模 1039 的主二次根。
12.6.07 求 892 模 1039 的主二次根。
12.6.08 求 3 模 107 的主二次根。
12.6.09 求 9 模 107 的主二次根。
12.6.10 求 4 模 107 的主二次根。
12.6.11 求 2 模 1 000 039 的主二次根。

12.7 mod p 的高次根

概括以上二次根的各种情况，在特殊条件下我们有求解模 p 的 n 次根的公式。如果存在整数 x，使得 $x^n = y \bmod p$，那么非零整数 y 就是模 p 的 n 次**幂**（或者，在早先的术语中称为 n 次**幂剩余**；如果没有这样的 x，那么 y 称为 n 次**幂非剩余**）。

定理 假设 p 是一个素数。如果 n 和 $p-1$ 互素，那么每一个 y 都有模 p 的 n 次根。特别的，假设 r 是 n 模 $p-1$ 的乘法逆元，那么 $y \bmod p$ 的一个 n 次根为 $y^r \% p$。

证明 我们来验证 $(y^r)^n = y \bmod p$。如果 $p \mid y$，则 $y = 0 \bmod p$，这显然。所以现在假设 p 不整除 y。由于 $rn = 1 \bmod p-1$，存在正整数 ℓ，使得 $rn = 1 + \ell(p-1)$，那么

$$(y^r)^n = y^{rn} = y^{1+\ell(p-1)} = y^1 \cdot y^{\ell(p-1)} = y \cdot (y^{(p-1)})^\ell = y \cdot 1^\ell = y \bmod p$$

这里用到了费马小定理 $y^{p-1} = 1 \bmod p$。 ♣

已经解决了 $gcd(n, p-1) = 1$ 的情况，我们将忽略 $1 < gcd(n, p-1) < n$ 这种情况，而关注 $gcd(n, p-1) = n$ 这一相对极端的情况：当 $n \mid (p-1)$ 时，只要 $gcd(n, \frac{p-1}{n}) = 1$ 成立，我们有一个有效的计算方法来求解模素数 p 的 n 次根。

定理 假设 p 是素数，且满足 $p = 1 \bmod n$，而且 $gcd(n, \frac{p-1}{n})$ 为 1。假设 r 是 n 模 $(p-1)/n$ 的乘法逆元。如果 y 是一个 n 次幂，那么 $y \bmod p$ 的一个 n 次根是 $y^r \% p$。

证明 这个命题的基本机制与前面定理的证明一样，有些复杂。首先我们来验证 $(y^r)^n = y \bmod p$。由于 $p \mid y$，所以 $y = 0 \bmod p$，这显然。所以现在假设 p 不整除 y。由于 $rn = 1 \bmod (p-1)/n$，存在正整数 ℓ，使得 $rn = 1 + \ell(p-1)/n$。而且，由于我们假设 y 有一个 n 次根，所以我们将 y 表达为 $y = x^n \bmod p$。那么

$$(y^r)^n = ((x^n)^r)^n = x^{n \cdot rn} = x^{n \cdot (1 + \frac{\ell(p-1)}{n})} = x^n \cdot x^{\ell(p-1)}$$
$$= x^n \cdot (x^{p-1})^\ell = x^n \cdot 1^\ell = x^n = y \bmod p$$

这里用到了费马小定理的结果 $y^{p-1} = 1 \bmod p$。 ♣

备注 如果 y 不是一个 n 次幂，那么后一个定理中的公式就不成立。当然，在计算之前并没有必要知道 y 是否是一个 n 次幂。更合适的是，应用这个公式，然后检验这个结果的 n 次幂是否是 y。如果是，那么 y 就是一个 n 次幂；如果不是 y，那么 y 就不是一个 n 次幂。

备注 如果 $gcd(n, \frac{p-1}{n}) > 1$，那么根的计算将变得更复杂。

备注 注意在上述两个定理中针对两种不同情况求 n 次根的两个公式之间的区别，这一点很重要。一旦出现混淆将产生没有意义的结果。

习题

12.7.01 求 2 模 101 的一个立方根。

12.7.02 求 46 模 101 的一个立方根。

12.7.03 求 97 模 101 的一个立方根。

12.7.04 求14模103的一个立方根。

12.7.05 求61模103的一个立方根。

12.7.06 求2模101的一个11次根。

12.7.07 求58模199的一个11次根。

第13章 模合数的根

现在，我们可以利用模数是素数的情况将一些问题化简，从而计算模数是合数的情况。对此的一个重要应用就是理解 $\mathbf{Z}/n$ 结构，这个结构中的区别在于 n 是否是素数。这一点关系到素性检验、因式分解攻击以及对许多公钥密码总体安全性的理解。

这种计算中一个值得关注的特性将出现在平方根 oracle 一节中。我们将看到对于两个模 4 同余 3 的素数 p,q，大整数 n 是它们的乘积，可以在给定（概率的）算法下通过寻找 $\bmod n$ 的二次根来求解 n 的因子 p,q。也就是说，平方根 oracle 可以应用于获得特定的因式分解。

本节的一些证明是不完整的，这是因为它们依赖于模素数的本原根的存在性，其存在性我们将稍后证明。但是，这些结果却阐明了它与其后一些更抽象结论之间的相关性。

作为费马的天才继承者，欧拉系统地发展了费马在数论领域的成果及其应用。正如可从名称所猜测到的欧拉定理，还有用来获取 $\bmod m$ 的 n 次根的欧拉判别准则，都要归功于欧拉。

13.1 孙子定理

孙子定理（Sun Ze's Theorem）有时被称为**中国剩余定理**，这主要是因为这个结果最早（由孙子以及孙子之后的学者发现）是在中国获得的。孙子的结果是在公元前 450 年得到的，以下的论述是秦九韶于 1250 年获得的。事实上通过相同的证明，这个结论可以应用于比整数 $\mathbf{Z}$ 更普遍的“数”。

假设 $m_1,\cdots,m_n$ 是非零整数，且使得对任意的指标对 i,j，如果 $i\neq j$，则 m_i 和 m_j 是互素的。我们就称整数 m_i 是**两两互素的**。如果

$$\mathbf{Z}/m_1\times\mathbf{Z}/m_2\times\cdots\times\mathbf{Z}/m_n$$

（通常）表示 n 元有序数组集合，其中第 i 个元素属于 $\mathbf{Z}/m_i$。

通过

$$f(x-\bmod-(m_1\cdots m_n))=(x-\bmod-m_1,x-\bmod-m_2,\cdots,x-\bmod-m_n)$$

定义一个映射

$$f:\mathbf{Z}/(m_1\cdots m_n)\to\mathbf{Z}/m_1\times\mathbf{Z}/m_2\times\cdots\times\mathbf{Z}/m_n$$

定理（孙子） $m_1,\cdots,m_n$ 两两互素，映射

$$f:\mathbf{Z}/(m_1\cdots m_n)\to\mathbf{Z}/m_1\times\mathbf{Z}/m_2\times\cdots\times\mathbf{Z}/m_n$$

是一一映射（双射）。

证明 首先，我们考虑只有两个互素的模数 m,n 的情况，通过

$$f(x-\bmod-mn)=(x-\bmod-m,x-\bmod-n)$$

得到相应的映射

$$f:\mathbf{Z}/mn\to\mathbf{Z}/m\times\mathbf{Z}/n$$

是一一映射。首先，我们证明这个映射是单射：如果 $f(x)=f(y)$，而且 $x\equiv y\bmod m$，$x\equiv y\bmod n$。也就是说，$m\mid x-y$，$n\mid x-y$。由于 m,n 是互素的，可以得到 $mn\mid x-y$，所以

$x \equiv y \bmod mn$。

从这点来说，因为 $\mathbf{Z}/mn$ 和 $\mathbf{Z}/m \times \mathbf{Z}/n$ 是有相同元素个数（也就是 mn）的有限集，任何单射同时也是满射。所以我们可以得出结论，f 是满射（因此是一一映射）。

但对满射性质做进一步的研究是有必要的。因为 m,n 互素，就存在整数 s,t，使得

$$sm + tn = 1$$

（如果需要的话，我们可以通过欧几里得算法计算出 s,t，但现在还不需要这样做）。然后我们对给定的整数 a 和 b，

$$f((b(sm) + a(tn)) - \mathrm{mod} - mn) = (a - \mathrm{mod} - m, b - \mathrm{mod} - n)$$

当然，

$$b(sm) + a(tn) \equiv b(sm) + a(1 - sm) \equiv a \bmod m$$

同样的，

$$b(sm) + a(tn) \equiv b(1 - tn) + a(tn) \equiv b \bmod n$$

这就证明了在两个模数的情况下，函数 f 的满射性，因此也就是一一映射。

现在我们考虑以任意的 $m_1, \cdots, m_n$（两两互素）为模的情况。我们对 n 进行归纳，我们已经证明了 $n=2$ 的情况，如果 $n=1$，结论是显然的。所以假设 $n>2$。通过对 n 进行归纳，由

$$f_0(x - \mathrm{mod} - m_2 \cdots m_n) = (x - \mathrm{mod} - m_2, x - \mathrm{mod} - m_3, \cdots, x - \mathrm{mod} - m_n)$$

定义的映射

$$f_0 : \mathbf{Z}/m_2 \cdots m_n \to \mathbf{Z}/m_2 \times \mathbf{Z}/m_3 \times \cdots \times \mathbf{Z}/m_n$$

是一个一一映射。因此由

$$f_1(x - \mathrm{mod} - m_1, x - \mathrm{mod} - m_2 \cdots m_n)$$
$$= (x - \mathrm{mod} - m_1, x - \mathrm{mod} - m_2, x - \mathrm{mod} - m_3, \cdots, x - \mathrm{mod} - m_n)$$

定义的映射

$$f_1 : \mathbf{Z}/m_1 \times \mathbf{Z}/m_2 \cdots m_n \to \mathbf{Z}/m_1 \times \mathbf{Z}/m_2 \times \mathbf{Z}/m_3 \times \cdots \times \mathbf{Z}/m_n$$

是一一映射。

与此同时，根据因式分解的惟一性，m_1 和乘积 $m_2 m_3 \cdots m_n$ 是互素的，所以，当 $n=2$ 时，由

$$f_2(x - \mathrm{mod} - m_1(m_2 \cdots m_n)) = (x - \mathrm{mod} - m_1, x - \mathrm{mod} - m_2 \cdots m_n)$$

定义的映射

$$f_2 : \mathbf{Z}/m_1(m_2 \cdots m_n) \to \mathbf{Z}/m_1 \times \mathbf{Z}/m_2 \cdots m_n$$

是一一映射。所以，复合映射

$$f = f_2 \circ f_1$$

也是一一映射的。 ♣

习题

13.1.01 证明 $x = 2 \bmod 6$ 且 $x = 3 \bmod 4$ 无解。

13.1.02 证明 $x = 2 \bmod 6$ 且 $x = 0 \bmod 4$ 有解。

13.2 特殊方程组

现在，我们按照同时求解几个同余式的方式来解释上一节的定理。这和线性方程组的初步讨论虽有一些类似之处，却有着关键性的区别。

在开始之前，我们来考查如下形式的最小非平凡方程组

$$\begin{cases} x \equiv a \bmod m \\ x \equiv b \bmod n \end{cases}$$

其中 m,n 互素，a,b 是任意整数，我们要找出所有满足这个条件的整数 x。

注意到这里有两个同余式却只有一个未知数，这在等式的情况下可能很容易就判断解是不存在的。但在同余方程组中就有少许区别，简化条件是：**我们只考虑模 m 和 n 是互素的情况，也就是说，** $gcd(m,n)=1$。

再次使用欧几里得算法，由于我们假设 $gcd(m,n)=1$，可以得到整数 s,t 使得

$$sm+tn=1$$

我们可以对上式重新进行整理，例如

$$tn=1-sm$$

这里我们就发现了一个窍门：单个同余式

$$x_0 = a(tn) + b(sm) \bmod mn$$

和上面的同余式方程组是**等价的**（有相同的解集）。

我们来检验一下：对模 m，我们有

$$\begin{aligned} x_0 &\equiv (a(tn)+b(sm)) \bmod m \equiv a(tn)+0 \bmod m \\ &\equiv a(tn) \bmod m \equiv a(1-sm) \bmod m \\ &\equiv a(1) \bmod m \equiv a \bmod m \end{aligned}$$

对同余模 n 的讨论是一样的。我们可以这样做：

$$\begin{aligned} x_0 &\equiv (a(tn)+b(sm)) \bmod n \equiv 0+b(sm) \bmod n \\ &\equiv b(sm) \bmod n \equiv b(1-tn) \bmod n \\ &\equiv b(1) \bmod n \equiv b \bmod n \end{aligned}$$

因此，任何满足模 mn 这个条件的 x_0 都是方程组的解。

另一方面，假设 x 是方程组的解，我们证明 x 和 x_0 模 mn 同余。由于 $x \equiv a \bmod m$ 和 $x \equiv b \bmod n$，我们有

$$x-x_0 \equiv a-a \equiv 0 \bmod m$$

以及

$$x-x_0 \equiv b-b \equiv 0 \bmod n$$

也就是说，m 和 n 都整除 $x-x_0$。由于 m 和 n 是互素的，我们可以得出结论，mn 整除 $x-x_0$。

注意到为了提高计算效率（不仅仅是理论上可行的），上述整个求解过程以及解的表示公式都使用了欧几里德算法。

例如，我们解方程组

$$\begin{cases} x \equiv 2 \bmod 11 \\ x \equiv 7 \bmod 13 \end{cases}$$

为了将这两个同余式结合在一起，我们对 11 和 13 使用欧几里得算法，可得

$$6\cdot 11-5\cdot 13=1$$

因此，利用已有的烦琐公式，单个同余式

$$x\equiv 2(-5\cdot 13)+7(6\cdot 11)\bmod 11\cdot 13$$

和给出的方程组是等价的。特别的，这给出了解

$$x\equiv -2\cdot 5\cdot 13+7\cdot 6\cdot 11\equiv 332\bmod 11\cdot 13$$

更一般地，考虑一个方程组

$$\begin{cases} x\equiv b_1 \bmod m_1 \\ x\equiv b_2 \bmod m_2 \\ x\equiv b_3 \bmod m_3 \\ \cdots \\ x\equiv b_n \bmod m_n \end{cases}$$

我们只考虑 m_i 和 m_j 互素的情况（其中 $i\neq j$）。我们分几个步骤来解答：首先，只考虑子方程组

$$\begin{cases} x\equiv b_1 \bmod m_1 \\ x\equiv b_2 \bmod m_2 \end{cases}$$

并且用上面的方法将方程组变为一个（等价的）同余式，形式是

$$x\equiv c_2 \bmod m_1m_2$$

然后考虑方程组

$$\begin{cases} x\equiv c_2 \bmod m_1m_2 \\ x\equiv b_2 \bmod m_3 \end{cases}$$

再次用上面的方法将方程组转化为一个等价的同余式，即

$$x\equiv c_3 \bmod m_1m_2m_3$$

依此类推。

备注 的确，这个过程只是对前一节定理证明的一个解释。

对这些形式特殊的线性同余方程组，对其求解方法可以用一个概念上更清晰的方式予以描述。

定理 假设 $m_1,\cdots,m_n$ 是两两互素的整数，考虑同余方程组

$$\begin{cases} x\equiv b_1 \bmod m_1 \\ x\equiv b_2 \bmod m_2 \\ x\equiv b_3 \bmod m_3 \\ \cdots \\ x\equiv b_n \bmod m_n \end{cases}$$

我们假设 M 是 $m_1,\cdots,m_n$ 的乘积，即 $M=m_1\cdots m_n$ 并假设 $M_i=M/m_i$。如果 $T_i=M_i^{-1} \bmod m_i$（由于假设中 M_i 和 m_i 互素，所以这是存在的），则

$$x=T_1M_1b_1+\cdots+T_nM_nb_n \bmod M$$

是这个方程组模 M 的惟一解。

证明 我们证明后一个公式，对所有的 i，求解 $x=b_i \bmod m_i$。当然，若 $j\neq i$，$m_i \mid M_j$，所以

$$T_j M_j b_j = T_j \cdot 0 \cdot b_j = 0 \bmod m_i \quad \text{对 } j \neq i$$

并且

$$T_i M_i = 1 \bmod m_i$$

也就是说，除了第 i 项外，其他所有直和项模 m_i 为 0，第 i 项模 m_i 为 b_i。因此，对所有的 i，这个表达式给出了模 m_i 的一个解。这个方程组中模 M 解的惟一性已经在孙子定理中证明，至此证明完成。 ♣

习题

13.2.01 求一个整数 x，使得 $x = 1 \bmod 12$ 且 $x = 1 \bmod 35$。

13.2.02 求一个整数 x，使得 $x = 1 \bmod 13$ 且 $x = 1 \bmod 36$。

13.2.03 求一个整数 x，使得 $x = 1 \bmod 12$ 且 $x = 8 \bmod 35$。

13.2.04 求一个整数 x，使得 $x = 5 \bmod 12$ 且 $x = 19 \bmod 35$。

13.2.05 求一个整数 x，使得 $x = 0 \bmod 12$ 且 $x = 1 \bmod 35$。

13.2.06 求一个整数 x，使得 $x = 1 \bmod 12$ 且 $x = 0 \bmod 35$。

13.2.07 求一个整数 x，使得 $x = 5 \bmod 48$ 且 $x = 3 \bmod 35$。

13.3 模是合数的同余方程

通常，在求解诸如 $x^2 \equiv b \bmod m$ 这类模 $m = m_1 m_2$（其中 m_1 和 m_2 是互素的）为合数的同余式时，分别求解模 m_1 和模 m_2 的同余式，然后再应用孙子定理将它们合并为模 m 的解，这样要比直接求解模 m 的解要快一些。如果 m 的素数分解是已知的，这一点就更为明显。

例如，我们手工求解方程

$$x^2 \equiv -1 \bmod 13 \cdot 17 \cdot 29$$

（由于我们并不希望手工求取所有 $13 \cdot 17 \cdot 29 = 6409$ 因数的可能性，所以用穷举法是不可取的）。通过观察孙子定理可以断言，满足 $x^2 \equiv -1 \bmod 6409$ 的模 6409 整数 x 的集合同三元组集合（x_1, x_2, x_3）存在一一对应关系，其中 $x_1 \in \mathbf{Z}/13$，$x_2 \in \mathbf{Z}/17$，$x_3 \in \mathbf{Z}/29$，并且

$$x_1^2 \equiv -1 \bmod 13 \qquad x_2^2 \equiv -1 \bmod 17 \qquad x_3^2 \equiv -1 \bmod 29$$

这里的一一映射是

$$x - \bmod - 6409 \to (x - \bmod - 13, x - \bmod - 17, x - \bmod - 29)$$

而且，上面的讨论揭示了在其他方面应该怎么做，也就是说，怎样由 $\mathbf{Z}/13 \times \mathbf{Z}/17 \times \mathbf{Z}/29$ 中解得到原来方程的解。

在这个例子中，13、17、29 这些数字并不是很大，对 −1 模 13、17 和 29 的平方根进行穷举求取也并不需要很长的时间。让我们描述一下在模 29 的情况下的查找过程。首先，$-1 = 28 \bmod 29$，但 28 不是一个平方数。下一步，给 28 加 29：57 不是平方数。给 57 加 29：86 不是平方数。给 86 加 29：115 不是平方数。给 115 加 29：$144 = 12^2$。因此，±12 是 $-1 \bmod 29$ 的平方根。类似的，我们发现 ±5 是 $-1 \bmod 13$ 的平方根，±4 是 $-1 \bmod 17$ 的平方根。

为了利用孙子定理从这几个结果中得到模 $6409 = 13 \cdot 17 \cdot 29$ 的一个解，我们首先要求整数 $s, t, s \cdot 13 + t \cdot 17 = 1$。有关最大公因子的理论结果保证存在这样的 s, t，并且可以利用欧几里得算法求出：

$$17 - 1 \cdot 13 = 4$$

$$13-3\cdot 4=1$$

反过来：

$$\begin{aligned}1&=13-3\cdot 4\\&=13-3\cdot(17-1\cdot 13)\\&=4\cdot 13-3\cdot 17\end{aligned}$$

因此，从 -1 模 13 的平方根 5 和 -1 模 17 的平方根 4，我们可以得到 -1 模 $13\cdot 17$ 的平方根：

$$4\cdot(4\cdot 13)-5\cdot(3\cdot 17)=-47 \bmod 13\cdot 17$$

继续下去，现在我们要求 s,t，使得

$$s\cdot(13\cdot 17)+t\cdot 29=1$$

应用欧几里得算法，注意到 $13\cdot 17=221$：

$$\begin{aligned}221-7\cdot 29&=18\\29-1\cdot 18&=11\\18-1\cdot 11&=7\\11-1\cdot 7&=4\\7-1\cdot 4&=3\\4-1\cdot 3&=1\end{aligned}$$

反过来，我们得到

$$\begin{aligned}1&=4-1\cdot 3\\&=4-1\cdot(7-1\cdot 4)\\&=2\cdot 4-1\cdot 7\\&=2\cdot(11-7)-7\\&=2\cdot 11-3\cdot 7\\&=2\cdot 11-3\cdot(18-11)\\&=5\cdot 11-3\cdot 18\\&=5\cdot(29-18)-3\cdot 18\\&=5\cdot 29-8\cdot 18\\&=5\cdot 29-8(221-7\cdot 29)\\&=61\cdot 29-8\cdot 221\end{aligned}$$

所以，从 -1 模 221 的平方根 -47 以及 -1 模 29 的平方根 12，我们得到平方根

$$12(-8\cdot 221)+(-47)(61\cdot 29)=-104359=4594 \bmod 6409$$

而且，由于我们在前面的步骤中已经得到了 s,t 在求解其他 -1 模 $13\cdot 17\cdot 29$ 的七个平方根，以上过程中最烦琐的部分就不再重复了。

习题

13.3.01 求一个整数 x，使得 $3x\equiv 2 \bmod 5$ 且 $4x\equiv 5 \bmod 7$。

13.3.02 求一个整数 x，使得 $7x\equiv 11 \bmod 5$ 且 $3x\equiv 22 \bmod 35$。

13.3.03 求模 $5\cdot 7$ 互不相同的四个整数，使得 $x^2\equiv 1 \bmod 5$ 且 $x^2\equiv 1 \bmod 7$。也就是说，求 1 模 35 的 4 个不同的平方根。

13.3.04 求模 $7\cdot 11$ 互不相同的四个整数，使得 $x^2\equiv 1 \bmod 7$ 且 $x^2\equiv 1 \bmod 11$。也就是说，

求1模77的4个不同的平方根。

13.3.05 求模13·17互不相同的四个整数，使得 $x^2 \equiv 1 \bmod 13$ 且 $x^2 \equiv 1 \bmod 17$。也就是说，求1模221的4个不同的平方根。

13.3.06 求2模7·23的四个不同的平方根。

13.3.07 解释为什么1模3·5·7=105有八个不同的平方根。

13.4 亨泽尔引理

在许多情况下，当 p 为素数时，求解多项式方程 $f(x) \equiv 0 \bmod p$ 能够保证模 p 的幂 p^n 存在解，而且也可以有效的求得这些解。除此之外，有趣的是整个求解过程与在微积分中牛顿对根的数值逼近方法非常类似。特别要提的是，我们将通过纯粹的泰勒展开式的代数形式来证明这个结果。

首先，我们给出一个数值的例子来说明亨泽尔（Hesel）引理所包含的思想。假设我们希望找到满足 $x^2 \equiv 2 \bmod 7^3$ 的 x。注意到 $\bmod 7^3$ 的一个解一定也是 $\bmod 7$ 的一个解，我们通过求 $\bmod 7$ 的一个解开始。由于在 $\mathbf{Z}/7$ 中只有7个元素，所以这更容易些，通过快速不断地试验我们看到 $(\pm 3)^2 = 9 = 2 \bmod 7$。

现在我们给出更进一步的求解方法：作为一个乐观主义者，我们设想可以通过向其增加（或减少）7的倍数，简单地调整 $3 \bmod 7$ 的解以获得一个 $\bmod 7^2$ 的解。也就是说，我们设想对某个 $y \in \mathbf{Z}$

$$(3 + 7 \cdot y)^2 \equiv 2 \bmod 49$$

展开，我们得到

$$9 + 21y + 49y^2 \equiv 2 \bmod 49$$

令人高兴的是，由于其系数被49所整除，y^2 项消失了（模49）。重新整理可得

$$7 + 42y \equiv 0 \bmod 49$$

用7去除这个式子则有

$$1 + 6y \equiv 0 \bmod 7$$

由于 y 的系数（也就是6）模7是可逆的，所以我们可求得 $y = (6^{-1})(-1) = 6 \cdot (-1) = 1 \bmod 7$ 的解。因此，

$$3 + 7 \cdot 1 = 10$$

是一个2模 7^2 的平方根。

继续进行乐观估计：为了得到 7^3 的解，现在我们希望可以通过增加 7^2 的倍数来调整 $10 \bmod 7^2$ 的解。也就是说，我们希望找到这样的 y，使得

$$(10 + 7^2 y)^2 \equiv 2 \bmod 7^3$$

展开并化简，得到

$$294y \equiv -98 \bmod 7^3$$

用 7^2 去除这个式子

$$6y \equiv -2 \bmod 7$$

$6 \bmod 7$ 的逆还是6，所以

$$y \equiv 6(-2) \equiv 2 \bmod 7$$

因此

$$10+7^2\cdot 2=108$$

满足

$$108^2\equiv 2 \bmod 7^3$$

这比使用穷举法直接求解 $2 \bmod 7^3$ 的平方根要快的多。

为给亨泽尔引理更一般的结论做准备，我们需要对多项式的**导数**给出一个纯代数的描述。也就是说，我们并不希望对需要的定义做任何限定。假设

$$f(x)=c_n x^n+\cdots+c_0$$

其系数属于 $\boldsymbol{Z}$。简单的定义另一个多项式 f' 为

$$f'(x)=nc_n x^{n-1}+(n-1)c_{n-1}x^{n-2}+\cdots+2c_2x+c_1+0$$

备注 当然，如果将这个导数定义为一个极限，可以用我们已知“正确”的公式来定义它。

命题 对系数属于 $\boldsymbol{Z}$ 的多项式 f,g，使用导数定义的纯代数形式，对 $r\in Z$，我们有

$$(rf)'=rf' \quad \text{(常数乘法法则)}$$
$$(f+g)'=f'+g' \quad \text{(加法法则)}$$
$$(fg)'=f'g+fg' \quad \text{(乘法法则)}$$
$$(f\circ g)'=f'\circ g\cdot g \quad \text{(链式法则)}$$

证明 尽管在此我们没有提到极限，但通过微积分我们知道这些结论是正确的。 ♠

定理 （亨泽尔引理） 假设 f 是一个多项式，其系数属于 $\mathbf{Z}$。p 是一个素数，如果对 $x_n\in\mathbf{Z}$ 满足

$$f(x_n)\equiv 0 \bmod p^n$$

其中 $n>0$。假设 $f'(x_n)\neq 0 \bmod p$。令 $f'(x_1)^{-1}$ 是一个整数，并且是 $f'(x_1)$ 模 p 的乘法逆元。那么

$$x_{n+1}=x_n-f(x_n)f'(x_1)^{-1}$$

满足

$$f(x_{n+1})\equiv 0 \bmod p^{n+1}$$

而且，从这个构造中可得

$$x_{n+1}=x_n \bmod p^n$$

特别的，对每一个指标，

$$x_n=x_1 \bmod p$$

备注 注意到 $f'(x_1)^{-1} \bmod p$ 这个数在迭代的每个循环中并不需要重复计算，只需要在开始时计算一次就可以了。

证明 首先，我们验证如果 f 的系数为整数，那么对每个正整数 k，商 $f^{(k)}/k!$ 的系数为整数，其中 $f^{(k)}$ 是 f 的 k 阶导数。证明这一点只需要考查 $f(x)=x^n$ 就足够了，因为每个整数系数多项式都是这种形式多项式的一个倍数的和。在这种情况下，$f^{(k)}/k!=\binom{n}{k}x^{n-k}$。由于 $\binom{n}{k}$ 是作为 $(x+1)^n$ 的系数，而 $(x+1)^n$ 的系数是整数，这就证明了我们的结论。

现在我们差不多可以证明这个定理了。假设 $y=-f(x_n)f'(x_n)^{-1} \bmod p^{n+1}$。注意到这个表达式使用了 $f'(x_n)^{-1} \bmod p^{n+1}$ 来替代 $f'(x_1)^{-1} \bmod p$。我们不得不从最后返回来考虑这个调整。

我们有 $y\equiv 0 \bmod p^n$。由于 f 是多项式，所以其任意一点的泰勒展开式都是有限的，并

且收敛于 f。因此

$$f(x_n+y)=f(x_n)+\frac{f'(x_n)}{1!}y+\frac{f''(x_n)}{2!}y^2+\frac{f^{(3)}(x_n)}{3!}y^3+\cdots$$

（这个和是有限的！）每个 $f^{(i)}(x_n)/i!$ 都是整数，而且 p^{2n} 整除 $y^2,y^3,y^4,\cdots$，所以

$$p^{2n}\text{ 整除 }\frac{f^{(2)}(x_n)}{2!}y^2+\frac{f^{(3)}(x_n)}{3!}y^3+\cdots$$

通过前面 y 的定义来替换这里的 y，我们有

$$f(x_n)+\frac{f'(x_n)}{1!}y=f(x_n)+\frac{f'(x_n)}{1!}(-f(x_n)f'(x_n)^{-1})$$

由于 $f'(x_n)^{-1}$ 是 $f'(x)$ 模 p 的一个乘法逆元，则存在整数 t，使得

$$f'(x_n)\cdot f'(x_n)^{-1}=1+tp$$

那么

$$f(x_n)+\frac{f'(x_n)}{1!}y$$

$$=f(x_n)+\frac{f'(x_n)}{1!}(-f(x_n)f'(x_n)^{-1})$$

$$=f(x_n)-f(x_n)(1+tp)=f(x_n)\cdot tp$$

因为 $f(x_n)\equiv 0 \bmod p^n$，我们已经得到 p 的另一个因子，也就是模 p^{n+1} 为 0。

而且，关于定理的最后一个结论，注意到我们利用 x_n 求 x_{n+1} 时用到的 $f(x_n)/f'(x_n)$ 是 p^n 的倍数。

最后，我们需要验证

$$f(x_n)f'(x_n)^{-1}=f(x_n)f'(x_1)^{-1} \bmod p^{n+1}$$

其中 $f'(x_1)^{-1}$ 只是 $\bmod p$ 的逆元，而不是 $\bmod p^{n+1}$ 的逆元。由于 $x_n=x_1 \bmod p$，而且 f' 的系数是整数，所以不难验证

$$f'(x_n)=f'(x_1) \bmod p$$

因此

$$f'(x_n)^{-1}=f'(x_1)^{-1} \bmod p$$

进一步，通过 p^n 整除 $f(x_n)$ 的假设，给上式左右两边都乘以 $f(x_n)$ 则得到

$$f(x_n)f'(x_n)^{-1}=f(x_n)f'(x_1)^{-1} \bmod p\cdot p^n$$

这就证明了我们不需要计算 $f'(x_n)^{-1} \bmod p^{n+1}$，而只需要计算 $f'(x_1)^{-1}$ 的结论。 ♣

备注 我们可以给出微分公式的纯代数证明并且通过其泰勒展开式表示多项式，不过这些在后面可以用更一般的方法做到，所以在这里我们仅限于基于微积分的讨论。

习题

13.4.01 通过亨泽尔引理求解 $\sqrt{2} \bmod 7^5$。

13.4.02 通过亨泽尔引理求解 $\sqrt{5} \bmod 11^6$。

13.4.03 通过亨泽尔引理求解 $\sqrt{3} \bmod 11^6$。

13.4.04 通过亨泽尔引理求解 $\sqrt{-1} \bmod 5^6$。

13.4.05 通过亨泽尔引理求解 $\sqrt{-1} \bmod 13^3$。

13.4.06(*) 讨论不能利用二次公式来求解方程 $x^2 + x + 1 \equiv 0 \bmod 2$ 的原因。

13.5 平方根 oracle

我们知道，oracle 是指可以回答某类问题的“黑盒子”，尽管我们并不需要知道他是怎样工作的。这个“oracle”的思想在讨论许多有关具有相对复杂性和可行性的计算问题时是重要的抽象概念，很自然它就会与有关密码安全性的问题联系在一起，而这些密码的安全则依赖不同问题的难度。这里我们将考虑一个特殊但很具代表性的 oracle。

定理 设 p,q 是两个模 4 同余 3 的大素数，$n = pq$，则一个可以求解模 n（并不知道其因子 p 和 q）的平方根的 oracle 可以用来获得一个求解 n 的素因子 p,q 的概率算法。

备注 例如，RSA 体制中的模数就是这样的结构。

备注 当然，我们希望看到这个算法的具体内容是什么，而不是仅仅知道它是存在的。因此，这个定理的证明包括算法的描述。

假设我们得到了上面的 $n = pq$，并且我们有一个寻找模 n 平方根的 oracle，但并不知道 n 的因子 p 和 q。我们重复下面的步骤：随机选取一 x，计算 $x^2 \% n$ 并将结果输入 oracle，就会得到一个 x^2 模 n 的平方根 y。

因为任何模素数的非零平方恰好有两个平方根，根据孙子定理，任何模 $n = pq$ 的平方都有 4 个平方根，$\pm x$ 是其中的两个。设另外两个是 $\pm x'$。假设最初 x 的选取是完全“随机的”，那么 oracle 得到的 y 是 $\pm x'$ 的概率为 1/2。如果得到的不是 $\pm x'$，则 n 不整除 $x \pm y$（因为 $y \neq \pm x \bmod n$），但 n 整除 $x^2 - y^2$（因为 $x^2 = y^2 \bmod n$）。所以 p 整除 $x \pm y$ 中的一个并且 q 整除另一个。因此 $gcd(x-y,n)$ 是 p 或 q（我们并不关心是哪一个）。这就完成了对概率算法单独一步的描述。

因为 oracle 可以被反复调用，每次调用都以 1/2 的概率获得一个因式分解。所以在 ℓ 次调用 oracle 后我们得到因式分解的概率是 $(1/2)^\ell$，随着 ℓ 的无限增大，这个概率迅速趋近于 0。例如，10 次调用才得到因式分解的概率是

$$\text{prob}(10\text{次调用}) = (1/2)^{10} \approx 0.001$$

这一般是不可能的。

实际上，在获得 n 的因式分解之前，我们可以计算期望调用 oracle 的次数：第一次调用得到因式分解的概率是 1/2。第一次尝试失败的概率是 1/2，并且第二次尝试成功的概率是 1/2，所以需要两次调用的概率是 1/4。类似的，两次都失败的概率是 $(1/2)^2$ 并且在第三次尝试成功的概率是 1/2，所以需要三次调用的概率是 1/8。通常地，前 $n-1$ 次都失败而在第 n 次成功得到一个因式分解的概率是 $(1/2)^n$。所以期望调用 oracle 的次数是

$$\text{期望调用 oracle 的次数} = \sum_{n=1}^{\infty} n \cdot (1/2)^n$$

我们可以通过生成函数计算这个式子：因为

$$x \cdot \frac{d}{dx} x^n = n \cdot x^n$$

所以在使用生成函数时，“通常”考虑

$$f(x)=\sum_{n=1}^{\infty} n\cdot x^n$$

我们注意到和为

$$f(x)=x\cdot\frac{d}{dx}\sum_{n=1}^{\infty}x^n$$

对几何级数进行求和为

$$\sum_{n=1}^{\infty}x^n=\frac{1}{1-x}(\text{因为}|x|<1)$$

我们最终得到

$$f(x)=x\cdot\frac{d}{dx}\frac{1}{1-x}=\frac{x}{(1-x)^2}$$

当 $x=1/2$ 时，计算得到 $f(1/2)=2$，所以得到因式分解所需要调用 oracle 的次数是 2。

例子 我们设 $p=103$ 和 $q=107$ 都模 4 同余 3，$n=pq=11021$。我们需要用到的 oracle 并不神秘，通过从费马小定理中导出的公式，将利用 p 和 q 的信息来计算模 n 的主平方根，然后利用孙子定理将它们组合。要建立一个 oracle，可以通过猜测加检验的方法，或者通过欧几里得算法，我们发现

$$(-27)\cdot 103+(26)\cdot 107=1$$

oracle 将通过公式

$$y \bmod p\text{的平方根}=y^{(p+1)/4} \bmod p=y^{26} \bmod 103$$

$$y \bmod q\text{的平方根}=y^{(q+1)/4} \bmod q=y^{27} \bmod 107$$

计算 $y \bmod p$ 和 $y \bmod q$ 的（主）平方根。然后用下面的公式将这两个结果组合

$$y \bmod n\text{的平方根}=-27\cdot y^{27}\cdot 103+26\cdot y^{26}\cdot 107 \bmod 11021$$

（当然，在计算这些方幂时我们可以随时根据需要利用模数 11021 进行约简）。

在利用概率算法尝试分解 n 时，需要选取一个“随机”整数 x：我们令 $x=50$。然后将 $50^2=2500$ 输入到 oracle 中，这将得到

$$-27\cdot(2500)^{27}\cdot 103+26\cdot(2500)^{26}\cdot 107 \bmod 11021=2625 \bmod 11021$$

根据 oracle 的输出，我们利用欧几里得算法来求相应的最大公因子 $gcd(2625-50\ 11021)$：

$$11\ 021-4\cdot 2575=721$$

$$2575-3\cdot 721=412$$

$$721-1\cdot 412=309$$

$$412-1\cdot 309=103$$

$$309-3\cdot 103=0$$

所以我们注意右边最后一个非零数，也就是 103，它确实是 11021 的一个因子。在这个例子中的 oracle 使得我们可以仅用一次尝试就得到了一个正确的因子。

习题

13.5.01(*) 如果 p,q,r 是互不相同的素数，$n=pqr$，并且我们有一个求解模 n 平方根的 oracle，你是否可以利用这个 oracle 求得一个分解 n 的概率算法？

13.5.02(*) 设 p 和 q 同余 4 mod 9 的两个“秘密”素数，并设 $n=p\cdot q$。假设我们有一个

求 mod n 立方根的 oracle。是否可以利用这个 oracle 求得一个分解 n 的概率算法？

13.6 欧拉定理

在这里，我们给出欧拉定理，它是对费马小定理的推广。欧拉定理的巧妙证明最好可作为基础群论的一个推论给出，但在这里我们还是只给出一个更简单的证明（解释少些）。

对于正整数 n，**欧拉函数** $\varphi(n)$ 是指满足如下条件的整数 b 的个数：$0<b<n$ 且 $gcd(b,n)=1$。

定理（欧拉） 对于和正整数 n 互素的 x，

$$x^{\varphi(n)} = 1 \bmod n$$

备注 对于素数 p，我们可以容易得到 $\varphi(p)=p-1$，所以 n 是素数这一特殊情况就是费马小定理。

证明 设 G 是所有模 n 有乘法逆元的整数的集合。首先我们注意乘积

$$P=\prod_{g\in G} g = G\text{中所有元素的乘积}$$

仍然属于 G。当然，设 $g_1,g_2,\cdots,g_t$ 是 G 中的所有元素，所以 $P=g_1g_2\cdots g_t$。那么我都应该知道 $P=g_1g_2\cdots g_t$ 的逆元就是每个因子逆元的乘积（并且顺序颠倒）：

$$(g_1g_2\cdots g_t)\cdot(g_t^{-1}\cdots g_2^{-1}g_1^{-1}) = 1 \bmod n$$

也就是说，尽管我们难以将它确定出来，但 P 确实有一个模 n 乘法逆元。

设 x 是 G 一元素。则我们将映射 $f:G\to G$ 定义为

$$f(g)=xg$$

这是 G 到自身的一个双射。首先，我们要验证 f 将 G 映射到自身：当然，对模 n 可逆的元素 x 和 g，

$$(xg)(g^{-1}x^{-1}) = 1 \bmod n$$

所以 $f(g)=xg$ 是 G 中元素。其次，验证单射性：如果 $f(g)=f(h)$，则由 f 的定义我们有 $xg=xh \bmod n$。等式左右同乘以 $x^{-1} \bmod n$，得到 $g=h \bmod n$，这就证明了单射性。最后，证明满射性：对给定 $g\in G$，我们能找到 $h\in G$ 使得 $f(h)=g$。就是说，能找到 $h\in G$ 使得 $xh=g$，所以 $h=x^{-1}g$ 是存在的。这就证明了双射性。

最后，我们进行讨论的计算部分。还是设 P 为 G 中所有元素的乘积，则

$$P=\prod_{g\in G} g = \prod_{g\in G} f(g)$$

由于映射 f 只是将 G 中的元素重新混合。则

$$P=\prod_{g\in G} f(g) = \prod_{g\in G} xg = x^{\varphi(n)}\prod_{g\in G} g = x^{\varphi(n)}\cdot P$$

其中 $\varphi(n)$ 是指 n 的欧拉函数。正如上面所证明的，因为 P 是模 n 可逆的，所以在等式左右同乘以 $P^{-1} \bmod n$ 可以得到

$$1 = x^{\varphi(n)} \bmod n$$

这就证明了欧拉定理。 ♣

备注 一方面，这里的讨论暗示存在一个更系统的通用方法。另一方面，存在其他同样重要的技术，而定理的简单证明中没有给出任何暗示。

习题

13.6.01 根据定义，求 $\varphi(30)$， $\varphi(15)$ 和 $\varphi(24)$。

13.6.02 根据定义，求 $\varphi(36)$， $\varphi(18)$ 和 $\varphi(28)$。

13.6.03 通过直接计算，验证2不是模17的原根，但3是。

13.7 原根的性质

在这一节中我们将简单的解释一个原根是什么样子的，并给出几个事实。模素数原根的存在性将于下面用来证明是否存在模素数的平方根（或 n 次根）的欧拉标准。对原根存在性（以及不存在性）的证明需要更多的准备。

设 n 是正整数，整数 g 是一个**模 n 的原根**，如果使得 $g^{\ell}=1 \bmod n$ 成立的最小正整数 λ 是 $\varphi(n)$。

注意到欧拉定理保证在任何情况下，跟 n 互素的整数 g 其指数 ℓ 不会大于 $\varphi(n)$，这是必要的。

对“大多数”整数 n 来说，没有模 n 的原根。关于何时存在或者不存在模 n 的原根更准确的陈述是

定理 存在模 n 原根的整数 n 总是具有如下形式：

- $n=p^e$， p 是奇素数，且 $e \geqslant 1$；
- $n=2p^e$， p 是奇素数，且 $e \geqslant 1$；
- $n=2,4$。

这将在稍后证明。特别要提的是，最重要的一点是确实存在模素数的原根。

理解有关原根如下一个重要性质是很有帮助的：

命题 设 g 是模 n 的一个原根，整数 λ 满足 $g^{\ell}=1 \bmod n$，则 $\varphi(n) \mid \ell$。

证明 根据除法性质，我们令 $\ell=q \cdot \varphi(n)+r$，其中 $0 \leqslant r<\varphi(n)$。则

$$1=g^{\ell}=g^{q \cdot \varphi(n)+r}=(g^{\varphi(n)})^{q} \cdot g^{r}=1^{q} \cdot g^{r}=g^{r} \bmod n$$

因为 g 是一个原根，$\varphi(n)$ 是使得 g 模 n 为1的最小指数。因此，由于 $1=g^{r} \bmod n$，所以一定有 $r=0$。也就是 $\varphi(n) \mid \ell$。 ♣

习题

13.7.01 证明2不是模23的原根。

13.7.02 证明3不是模23的原根。

13.7.03 已知2不是模23的原根，证明4不是模23的原根。

13.7.04 根据定义，计算 $\varphi(11)$。

13.7.05 根据定义，计算 $\varphi(13)$。

13.7.06 根据定义，计算 $\varphi(10)$。

13.7.07 根据定义，计算 $\varphi(12)$。

13.7.08 根据定义，计算 $\varphi(14)$。

13.7.09 根据定义，计算 $\varphi(24)$。

13.7.10 根据定义，计算 $\varphi(101)$。

13.7.11 根据定义，计算 $\varphi(103)$。

13.8 欧拉判别准则

这个判别准则可以有效地判断当 $p \equiv 1 \bmod n$ 时，整数 y 是否是模素数 p 的 n 次根。相反地，我们已经看到当 $gcd(n, p-1) = 1$ 时，所有的数都是模素数 p 的 n 次根。对平方根的情况可以进行更多的讨论，这在其后二次符号和二次互反定理的讨论中非常重要。（这里假设我们熟知快速指数算法。）为了证明欧拉判别准则，我们必须承认模素数**原根**的存在性，这将在稍后证明。

对给定的 y，y 模 m（m 不一定是素数）的**平方根**是指存在整数 x，使得

$$x^2 \equiv y \bmod m$$

如果有这样的 x 存在，则 y 是**模 m 的一个平方**，或者在以前的术语中称为**模 m 的二次剩余**。如果不存在这样的 x，则 y 是**模 m 的一个非平方**，或者称为**模 m 的二次非剩余**。

备注 和乘法逆元一样，可能存在于实数或复数中的这些平方根之间本质上没有必然的联系。因此，表达式“$\sqrt{y}$”或“$y^{1/2}$”对 $y \in \mathbf{Z}/m$ 没有本质区别。

例子 因为 $2^2 = 4 = -1 \bmod 5$，所以 2 是 -1 模 5 的一个平方根。我们记作

$$2 = \sqrt{-1} \bmod 5$$

注意，实数中不存在 -1 的平方根这一事实跟 -1 模 5 平方根的存在性并不矛盾。

例子 因为 $4^2 = 16 = 5 \bmod 11$，所以

$$4 = \sqrt{5} \bmod 11$$

例子 不存在满足 $\sqrt{2}$ 模 5 的数：为了证实这一点，我们计算如下 5 种情况：

$$0^2 = 0 \neq 2 \bmod 5$$

$$1^2 = 1 \neq 2 \bmod 5$$

$$2^2 = 4 \neq 2 \bmod 5$$

$$3^2 = 9 = 4 \neq 2 \bmod 5$$

$$4^2 = 16 = 1 \neq 2 \bmod 5$$

因为 $\mathbf{Z}/5$ 是由 5 个同余类 $\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}$ 所组成，所以我们不需要更多的检验就可以通过穷举知道 2 模 5 的平方根。

从简单的角度来看，似乎只有通过穷举才能验证是否存在 y 模 m 的平方根，将 $\mathbf{Z}/m$ 中的每个元素平方，看是否和 y 相等。从这个角度看，因为 $\mathbf{Z}/m$ 中所有元素都要试一遍，所以要确定一个数有没有平方根是非常费力的。在这里，费马小定理（连同快速指数算法）提供了一些便利，但目前我们只能证明下面定理的一半。

非零数 y 模 p 是模 p 的一个 **n 次幂**（或者在以前的术语中称为 **n 次剩余**），如果存在 x 使得 $x^n \equiv y \bmod p$。（如果不存在这样的 x，则称 y 是 **n 次非剩余**。）

定理 （欧拉判别准则）设 p 为素数，$p \equiv 1 \bmod n$。设 y 与 p 互素，则当且仅当 $y^{(p-1)/n} \equiv 1 \bmod p$ 时，y 是 $\bmod p$ 的 n 次幂。作为一个特殊情况，若 p 是奇素数，且 p 不整除 y，则当且仅当 $y^{(p-1)/2} \equiv 1 \bmod p$ 时，y 是 $\bmod p$ 的一个非零平方。

备注 这是对不是模 p 的平方的一个合理的测试。很明显，欧拉是三百年前第一个观察到这一点的人。后来，二次互反律给出了测试是否存在模 p 的平方的另一个机制。n 次幂准则不存在简单的替代物。

证明 容易部分：设 $y = x^n \bmod p$。则根据费马小定理，

$$y^{(p-1)/n} = (x^n)^{(p-1)/n} = x^{p-1} = 1 \bmod p$$

复杂部分：现在假设 $y^{(p-1)/n} = 1 \bmod p$，证明 y 是 $\bmod p$ 的 n 次幂。设 g 是一个模 p 本原根，ℓ 是一个整数，$g^\ell = y$。我们有

$$(g^\ell)^{(p-1)/n} = 1 \bmod p$$

从前面对本原根的讨论，我们知道

$$(p-1) \mid \ell \cdot (p-1)/n$$

（因为当 p 为素数时 $\varphi(p) = p-1$）。根据普通整数分解的惟一性，发生这种情况的惟一可能就是 ℓ 被 n 整除，就是存在整数 k，使得 $\ell = kn$。则

$$y = g^\ell = g^{kn} = (g^k)^n \bmod p$$

也就是说 y 是 g^k 的 n 次幂。 ♣

推论（欧拉判别准则） 设 p 是奇素数，y 和 p 互素，则

$$y^{(p-1)/2} \equiv 1 \bmod p \quad \text{如果}y\text{是} \bmod p\text{的平方}$$

或者

$$y^{(p-1)/2} \equiv -1 \bmod p \quad \text{如果}y\text{是} \bmod p\text{的非平方}$$

证明 需要证明的是，如果 y 是一个非平方，则 $y^{(p-1)/2} = -1 \bmod p$。根据费马小定理有 $y^{(p-1)} = 1 \bmod p$，一定有

$$(y^{(p-1)/2})^2 = 1 \bmod p$$

也就是说，$y^{(p-1)/2} \bmod p$ 满足 $x^2 = 1 \bmod p$，并且不等于1。可以肯定 -1 是 $x^2 = 1 \bmod p$ 的另一个解。如果我们能证明这个方程除了 ± 1 外没有其他的解，那就证明了结论。假设 x 是满足 $x^2 = 1 \bmod p$ 的一个整数，则根据定义，$p \mid (x-1)(x+1)$。

我们回想一下，如果 p 是素数且 $p \mid ab$，则有 $p \mid a$ 或 $p \mid b$。重温一下其中的原因是有好处的：假设 $p \mid ab$ 但 p 不整除 a，目标是证明 $p \mid b$。设 $ab = kp$，因为 p 是素数，所以 $gcd(a,p) = 1$，则存在整数 s,t 使得 $sp + ta = 1$。那么

$$b = b \cdot 1 = b \cdot (sp + ta) = bsp + tab = bsp + tkp = p \cdot (bs + tk)$$

因此有 $p \mid b$。

所以，如果 $p \mid (x-1)(x+1)$，则有 $p \mid (x-1)$ 或者 $p \mid (x+1)$，也就是说 $x = \pm 1 \bmod p$。这就完成了当且仅当 y（且和 p 互素）是 $\bmod p$ 的非平方时，$y^{(p-1)/2} = -1 \bmod p$ 的证明。 ♣

习题

13.8.01 2 是模101的平方吗？

13.8.02 14 是模101的平方吗？

13.8.03 69 是模101的平方吗？

13.8.04 2 是模109的平方吗？

13.8.05 5 是模109的立方吗？

13.8.06 69 是模109的立方吗？

13.8.07 2 是模109的立方吗？

13.8.08 68 是模109的立方吗？

13.8.09 105 是模1009的144次幂吗？

第14章 弱乘法性

14.1 弱乘法性的定义

许多在数论中出现的函数以及数学的其他部分都有一个特性，该特性被称为**弱乘法性**（下面定义）。这个思想借用自身以简化许多计算和证明。作为一个特殊和重要的例子，我们将证明根据 n 的素数分解来计算欧拉函数 $\varphi(n)$ 的公式。

设 f 是关于正整数的一个函数。这个函数可以是复数值的，也可以是简单整数值的：取值的类型并不重要。这种函数具有**乘法性**，如果总是有

$$f(mn)=f(m)\cdot f(n)$$

这被证明是一个强条件，在数论中出现的问题中应用起来也太严格，尽管这个条件就其本身而言是重要的。因此，我们将这个条件减弱，称 f 具有**弱乘法性**，如果

$$f(mn)=f(m)\cdot f(n) \qquad m,n\text{互素}$$

值得注意的是由于 $m=1$ 和 $n=1$ 是互素的，所以弱乘法性函数 f 对1的值 $f(1)$ 是 0 或 1，我们计算

$$f(1)=f(1\cdot 1)=f(1)\cdot f(1)$$

所以 $f(1)$ 是方程

$$x=x^2$$

的一个解，它的解只有 0,1 。（这个讨论将弱乘法性函数的取值应用于整数、有理数、实数或复数。大多数的特殊值应用起来需要对结论作修改。）

最标准的（相当基础的）弱乘法性函数族（不是真正的乘法性）是**因子方幂和**的函数

$$\sigma_0(n)=s_0(n)=n\text{的正因子个数}$$
$$\sigma_1(n)=s_1(n)=n\text{的正因子的和}$$
$$\sigma_k(n)=s_k(n)=n\text{的正因子的}k\text{次幂的和}$$

（实际上，指标 k 可以是任何复数。）

s_k 有弱乘法性的证明依赖于整数的惟一因子分解。设 m,n 互素。关键问题在于，由于因子分解惟一性以及 m 和 n 是互素的，所以乘积 mn 的每个正因子分别是 m 的正因子 d 和 n 的正因子 e 的乘积 de，而且可以惟一的表述为这种形式。所以

$$s_k(mn)=\sum_{D|mn,D>0}D^k=\sum_{d|m,e|n,d,e>0}(de)^k=\sum_{d|m,d>0}d^k\sum_{e|n,e>0}e^k=s_k(m)\cdot s_k(n)$$

另一个重要的弱乘法性函数是**墨比乌斯函数** μ，定义为

$$\mu(n)=\begin{cases}(-1)^\ell & \text{如果}n\text{有}\ell\text{个互不相同的素因子}\\ 0 & \text{如果}n\text{有平方因子}\end{cases}$$

弱乘法性是惟一因子分解的一个结果。

欧拉函数是一个比较精巧的例子：

$$\varphi(n)=\text{和}n\text{互素的整数}i\text{的个数并且}0<i\leqslant n$$

命题 欧拉函数有弱乘法性。也就是说，对 $gcd(m,n)=1$ 有

$$\varphi(mn)=\varphi(m)\varphi(n)$$

证明 根据 gcd 理论方面的定义，因为从表达式 $ay+bm=1$ 我们看到 a 是整数 y 模 m 的一个乘法逆元，所以，当且仅当 $gcd(y,m)=1$，y 有一个模 m 的乘法逆元。因此，

$$\varphi(m)=\mathbf{Z}/m\text{中有乘法逆元的元素个数}$$

而且，y 模 m 是乘法可逆的意思是同余式

$$xy\equiv 1 \bmod m$$

有解。孙子定理说明当 $m=m_1m_2$ 将 m 分解为两个互素的因子 m_1,m_2 时，$xy\equiv 1 \bmod m$ 的可解性等价于同余方程组

$$\begin{cases} xy\equiv 1 \bmod m_1 \\ xy\equiv 1 \bmod m_2 \end{cases}$$

的可解性。由于 $\varphi(m_1)$ 和 $\varphi(m_2)$ 分别是上述两个方程分别在 $\mathbf{Z}/m_1$ 和 $\mathbf{Z}/m_2$ 解的个数，所以我们有

$$\varphi(m_1m_2)=\varphi(m_1)\cdot\varphi(m_2)$$

其中 m_1 和 m_2 是互素的。 ♣

推论 设 $p_1<p_2<\cdots<p_k$ 是素数并且 $e_1,e_2,\cdots,e_k$ 是正整数。则

$$\varphi(p_1^{e_1}\cdots p_k^{e_k})=(p_1-1)p_1^{e_1-1}(p_2-1)p_2^{e_2-1}\cdots(p_k-1)p_k^{e_k-1}$$

证明 因为我们知道 φ 具有弱乘法性，只要证明对素数 p 和正整数 e，如下等式成立即可

$$\varphi(p^e)=(p-1)p^{e-1}$$

在这种情况下，$\varphi(p^e)$ 就是所有满足 $0<\ell\leqslant p^e$ 且和 p 互素的整数 ℓ 的个数。由于 p 是素数，这意味着 p 不整除 ℓ。计算这个范围中能被 p 整除的整数 ℓ 的数量是容易的：其中有 $1/p$ 个能被 p 整除，所以总数就为 p^e/p。因此，除去这部分可得

$$\varphi(p^e)=p^e-p^e/p=p\cdot p^{e-1}-p^{e-1}=(p-1)p^{e-1}$$

这就证明了我们的结论。 ♣

习题

14.1.01 计算1000的正因子的个数。

14.1.02 计算 $999=3\cdot3\cdot3\cdot37$ 的正因子的个数。

14.1.03 计算1000的正因子的和。

14.1.04 计算 $999=3\cdot3\cdot3\cdot37$ 的正因子的和。

14.1.05 计算 1 000 000 因子的平方和。

14.1.06 计算 10^{100} 的因子的个数。

14.2 算术卷积

这里我们关注一个问题，即由一个简单的弱乘法性函数来构造出一个更复杂的弱乘法性函数。这种（**算术**）**卷积**在数论中经常出现。

命题 设 f 是一个关于正整数的弱乘法性函数。定义另一个函数 F 为

$$F(n)=\sum_{d|n,d>0} f(d)$$

则 F 同样具有弱乘法性。

证明 设 m,n 互素，则

$$F(mn)=\sum_{D|mn,D>0} f(D)=\sum_{d|m,d>0,e|n,e>0} f(d\cdot e)$$

因为 mn 的正因子可以惟一的表示为乘积 de，其中 d 是 m 的正因子，e 是 n 的正因子。由于 $gcd(m,n)=1$，则 d 和 e 肯定是互素的。所以，$f(de)=f(d)f(e)$ 并且

$$f(de)=f(d)f(e)=\sum_{d|m,d>0} f(d)\sum_{e|n,e>0} f(e)=F(m)\cdot F(n)$$

这就证明了命题。 ♣

上面这个命题的构造是一个由已知的弱乘法性函数得到新弱乘法性函数的标准过程。对上述构造过程的一般化就是**卷积**：

命题 设 f 和 g 是关于正整数的弱乘法性函数，定义另一个函数 F，有时也称为 f 和 g 的**算术卷积**

$$F(n)=\sum_{d|n,d>0} f(d)g\left(\frac{n}{d}\right)$$

则 F 同样具有弱乘法性。

证明 设 m,n 互素，则

$$F(mn)=\sum_{D|mn,D>0} f(D)g\left(\frac{mn}{D}\right)=\sum_{d|m,d>0,e|n,e>0} f(d\cdot e)g\left(\frac{mn}{ed}\right)$$

因为 mn 的正因子可以惟一的表示为乘积 de，其中 d 是 m 的正因子，e 是 n 的正因子。而且由于 m 和 n 是互素的，那么 m/d 和 n/e 互素。由 $gcd(m,n)=1$，则 d 和 e 肯定是互素的。因此，$f(de)=f(d)f(e)$ 且 $g(mn/de)=g(m/d)g(n/e)$。所以，$F(mn)$ 的表达式变为

$$=\sum_{d|m,d>0} f(d)g(m/d)\sum_{e|n,e>0} f(e)g(n/e)=F(m)\cdot F(n)$$

这就证明了命题。 ♣

备注 第一个命题是第二个命题当 $g(n)=1$ 时（对所有的 n）的一个特殊情况，第二个命题不仅具有弱乘法性，而且还有普通乘法性。

习题

14.2.01 设 $f(n)$ 表示 n 的正因子的数量，$g(n)$ 表示 n 的正因子的和。设

$$h(n)=\sum_{0<d|n} f(d)g(n/d)$$

（d 的范围是 n 的所有正因子），计算 $h(1000)$。

14.2.02(*) 你是否能对上个练习中的函数 h 进行更基本或直观的表达？

14.3 墨比乌斯反演

这里的墨比乌斯反演公式是一个特殊类型的公式，它在整个初等数论中也是容易得出的。其他形式的墨比乌斯反演则贯穿于数学的各个方面。我们将它用于证明在稍后非常有用的欧拉函数的特殊性质。请记住 $\mu(n)$ 表示 n 的墨比乌斯（Möbius）函数。

定理 设 f 是关于正整数的弱乘法函数，并且

$$F(n)=\sum_{d|n,d>0} f(d)$$

则

$$f(n)=\sum_{d|n,d>0} F(d)\mu\left(\frac{n}{d}\right)$$

对称地，如果我们设

$$G(n)=\sum_{d|n,d>0} f(d)\mu\left(\frac{n}{d}\right)$$

则我们有

$$f(n)=\sum_{d|n,d>0} G(d)$$

证明 首先考虑对 F 的结论，通过将表达式看做 $f(n)$ 在任何情况下都是 F 和 μ 的卷积，可以将之简化。利用上面的结果，因为 f 具有弱乘法性，所以第一个式子中的 F 具有弱乘法性而且由于 μ 具有弱乘法性，则 F 和 μ 的卷积也具有弱乘法性。所以要证明 F 和 μ 的卷积等于弱乘法性函数 f 只需要考虑 $n=p^e$ 的情况，其中 p 是素数且 e 非负整数。最简单的情况是

$$F(1)=\sum_{d|1,d>0} f(d)=f(1)$$

并且

$$\sum_{d|1,d>0} F(d)\mu\left(\frac{1}{d}\right)=F(1)\mu(1)=F(1)=f(1)$$

所以我们已经证明了 $n=1$ 时关于 F 的结论。

做了足够的简化后开始计算：设 e 是正整数，p 是素数：

$$\sum_{d|p^e,d>0} F(d)\mu\left(\frac{p^e}{d}\right)=\sum_{0\leqslant i\leqslant e} F(p^i)\mu(p^{e-i})$$

根据其定义，

$$\mu(p^{e-i})=\begin{cases}1 & e-i=0\\ -1 & e-i=1\\ 0 & e-i>1\end{cases}$$

因此，对 i 的值求和实际只有两个值 $i=e-1$ 和 $i=e$，而且整个式子进一步简化为

$$\sum_{0\leqslant i\leqslant e} F(p^i)\mu(p^{e-i})=\sum_{i=e-1,e} F(p^i)\mu(p^{e-i})=\begin{cases}-F(p^{e-1})+F(p^e) & e>0\\ F(1) & e=0\end{cases}$$

现在根据定义用 f 代替 F：

$$=-\sum_{d|p^{e-1},d>0} f(d)+\sum_{d|p^{e},d>0} f(d)$$

现在注意到在第二项和式中，没有被第一项和式消去的部分只有 $d=p^e$ 这一项。所以，也就是说

$$-F(p^{e-1})+F(p^e)=-\sum_{d|p^{e-1},d>0} f(d)+\sum_{d|p^{e},d>0} f(d)=f(p^e)$$

这证明了两个弱乘法性函数对素数的方幂 p^e 成立，所以对其他情况同样成立。

对于 G，这个讨论除了在某些细节上有区别外都是类似的，所以我们也给出证明过程。跟前面一样，由于 f 和 G 都具有弱乘法性，还有 G 是 f 和 μ 的卷积。同样的，函数

$$H(n)=\sum_{d|n,d>0} G(d)$$

具有弱乘法性。为了证明两个弱乘法性函数 H 和 f 是相等的，只需要证明对素数方幂它们是相同的即可。设 p 是素数且 e 是非负整数，则

$$G(p^e)=\sum_{d|p^{e},d>0} f(d)\mu(\frac{p^e}{d})=\sum_{0\leqslant i\leqslant e} f(p^i)\mu(p^{e-i})$$

除非 $e-i$ 的值为 0 或 1， 否则 μ 得到的值为 0，所以也就是说

$$G(p^e)=\begin{cases} f(p^e)-f(p^{e-1}) & e>0 \\ f(1) & e=0 \end{cases}$$

则

$$H(p^e)=\sum_{d|p^{e},d>0} G(d)=G(1)+\sum_{d|p^{e},d>1} G(d)=G(1)+\sum_{1\leqslant i\leqslant e} G(p^i)=f(1)+\sum_{1\leqslant i\leqslant e}(f(p^i)-f(p^{i-1}))$$

$$=f(1)+(f(p)-f(1))+(f(p^2)-f(p))+\cdots+(f(p^e)-f(p^{e-1}))=f(p^e)$$

至此定理得证。 ♣

推论 对正整数 n，欧拉函数满足

$$\sum_{d|n,d>0}\varphi(d)=n$$

证明 注意到关于正整数的函数 $f(n)=n$ 有弱乘法性。根据墨比乌斯反演，这个推论的结论等价于

$$\varphi(n)=\sum_{d|n,d>0} f(d)\mu\left(\frac{n}{d}\right)$$

而且，根据所知道的弱乘法性质，只需要证明后一个等式在 $n=p^e$ 的情况即可，其中 p 是素数且 e 是正整数（$n=1$ 的情况可以直接验证）。也就是说，我们希望验证

$$\varphi(p^e)=\sum_{0\leqslant i\leqslant e} f(p^i)\mu(p^{e-i})$$

同前面一样，除非 $e-i$ 的值为 0 或 1，否则 $\mu(p^{e-i})=0$，所以这可以简化为

$$\varphi(p^e)=f(p^e)-f(p^{e-1})=p^e-p^{e-1}$$

当然， p^e 是满足 $0<\ell\leqslant p^e$ 的整数 ℓ 的数量，并且 p^{e-1} 是这个范围内能被 p 整除的 ℓ 的数量，所以它们的差就是与 p 互素的 ℓ 的数量。这就证明了推论。 ♣

习题

14.3.01 设 μ 是墨比乌斯函数，计算 $\mu(100)$。

14.3.02 设 μ 是墨比乌斯函数，计算 $\mu(144)$。

14.3.03 设 μ 是墨比乌斯函数，计算 $\mu(35)$。

14.3.04 设 μ 是墨比乌斯函数，计算 $\mu(105)$。

14.3.05 设 μ 是墨比乌斯函数，计算 $\mu(1\ 000\ 000)$。

14.3.06 设 μ 是墨比乌斯函数，计算 $\mu(d)$ 的和，d 取遍 10^{20} 所有正因子。

第15章 二次互反定理

这里有关“二次符号”的快速计算算法是许多算法的基础。也许第二重要的要称欧几里得算法了，这是我们所拥有的另一个好算法。重要的是，注意在判定给定的整数 b 是否模素数 p 的平方时，这里介绍的算法要比欧拉判别准则速度快的多。

即使从一个非计算的角度来看，由于二次互反定理将两个没有明显关系的事情联系起来，因此它的作用依然很大。但有时二次互反定理在计算上的重要性却被忽视了。

然而，有关二次符号如何应用的证明过程对我们而言有些复杂，所以在此省略了。

15.1 二次根

求解模 m 的方程跟求解有理数、实数或复数域上的方程在方法上有许多共同的特点。然而，为了使我们的直观认识在更一般的情况下仍然正确，需要对这些在处理实数和复数时得到的通用性不强的直观认识进行更进一步的精练。

对给定的模数 n 和数 b，问题是 b 是否有模 n 的一个二次根。也就是说，方程

$$x^2 \equiv b \bmod n$$

是否有解？

例如，

$$x^2 \equiv 2 \bmod 7$$

有解 $x = 3,4$，这是因为

$$3^2 = 9 \equiv 2 \bmod 7$$

$$4^2 = 16 \equiv 2 \bmod 7$$

另一方面，作为一个不同的例子，同余式 $x^2 \equiv 5 \bmod 7$ 就没有解，这可以通过引入 $0,1,2,3,4,5,6$（最坏的情况）进行验证，你将看到这些数的平方都不满足模 7 同余 5。

注意到上述的解和这样一个事实是没有关系的：即不存在整数 x，使得 $x^2 = 2$。而且这些解跟在实数域上近似于 2 的一个平方根也没有什么关联：

$$\sqrt{2} = 1.4142135623731\cdots$$

实际上，从完全直观的角度看，模 m 的平方根是否存在确是一件不容易说明的问题。至少我们早先有关实数或复数的二次根的经验看上去与此没有什么联系。

作为另一个例子，考虑

$$x^2 = -1 \bmod 5$$

到现在我们应该可以回答，实数的平方不可能是负数，所以整数的平方肯定不是负数（或者类似的）。然而，2 和 3 都是 $-1 \bmod 5$ 的二次根：

$$2^2 = 4 = -1 \bmod 5$$

$$3^2 = 9 = -1 \bmod 5$$

从一个相对直观的角度看，判断 $x^2 = c \bmod m$ 是否有解的惟一方法就是不断地试验，将

$1,2,\cdots,m-1$ 分别平方，然后看是否为 $c \bmod m$。这个方法至少有一个优点就是如果有二次根存在，那么我们不仅能知道其存在性，还能明确的找到它。

对于素数模数 p 且同余 $3 \bmod 4$，我们已经得到了求解二次根的公式

$$y \bmod p \text{ 的二次根} = y^{(p+1)/4} \bmod p$$

尽管确实存在一个求解二次根的概率算法，但对任意的素数我们都没有公式可用。要利用这些公式或算法，首先需要知道模数的因式分解。到现在，我们会意识到这可能是很难做到的。

15.2 二次符号

这里我们引入标准的记号，并将问题稍作改变以更适合二次互反定理。

对奇素数 p，**二次符号**或者**勒让德符号** $(b/p)_2$（有时简单的记为 (b/p)，省略了下标）定义为

$$\left(\frac{b}{p}\right)_2 = \begin{cases} 0 & \text{如果} gcd(b,p) > 1 \\ 1 & \text{如果} x^2 = b \bmod p \text{存在解} x \text{且} gcd(b,p) = 1 \\ -1 & \text{如果} x^2 = b \bmod p \text{无解且} gcd(b,p) = 1 \end{cases}$$

注意到

$$\text{如果 } b \equiv b' \bmod p \quad \text{那么} \left(\frac{b}{p}\right)_2 = \left(\frac{b'}{p}\right)_2$$

备注 为了强调这个符号，我们将在二次符号中使用下标“2”，即使我们不会使用其他类似 $(x/p)_r$，$r \neq 2$ 的符号。这里的其他符号表示 x 是否为模素数 p 的 r 次幂。我们不会用到这些，但它们比二次符号显得更精巧。

那么对任意整数 n，其因式分解为

$$n = 2^{e_0} p_1^{e_1} \cdots p_k^{e_k}$$

其中 p_i 是奇素数，定义**扩展二次符号或雅可比符号**为

$$\left(\frac{b}{n}\right)_2 = \left(\frac{b}{p_1}\right)_2^{e_1} \cdots \left(\frac{b}{p_k}\right)_2^{e_k}$$

对素数 p，二次（勒让德）符号 $(b/p)_2$ 是指 b 是否为模 p 的一个二次根。这就是整个讨论的原因所在。然而，这不包括非素数 n 的情况。也就是说，如果 n 不是素数，那么这个（雅可比）符号 $(b/n)_2$ 所得到的值并不能直接表明 $x^2 = b \bmod n$ 是否有解。

因此，雅可比符号更多的起到一个**辅助的**作用，在勒让德符号计算中很有用。这项辅助作用的效用归因于使用二次互反律可以快速计算雅可比符号这个事实。

15.3 乘法性质

如果我们承认欧拉判别准则可以判别一个数是否为模素数的平方，那么我们可以证明二次符号重要的基本乘法性质。我们在早先曾经陈述过下面这个定理：

定理（欧拉判别准则） 如果素数 $p > 2$，对整数 b 有

$$\left(\frac{b}{p}\right)_2 \equiv b^{(p-1)/2} \bmod p$$

推论

$$\left(\frac{-1}{p}\right)_2 = (-1)^{(p-1)/2}$$

（这是定理的一个特殊情况，它利用了在对 -1 求幂时我们不需要用模 p 进行约简这一事实）

现在我们有了二次符号的**乘法性质**：

推论 对素数 p 和整数 a,b，

$$\left(\frac{ab}{p}\right)_2 = \left(\frac{a}{p}\right)_2 \cdot \left(\frac{b}{p}\right)_2$$

证明 利用上述定理，这个推论就不难证明：计算模 p

$$\left(\frac{ab}{p}\right)_2 = (ab)^{(p-1)/2} = a^{(p-1)/2} b^{(p-1)/2} = \left(\frac{a}{p}\right)_2 \cdot \left(\frac{b}{p}\right)_2 \bmod p$$

备注 这个公式至少说明，如果 a 和 b 都是模素数 p 的非平方，那么它们的乘积 ab 就是 mod p 的平方。

乘法性质更一般的形式是：

推论 对任意的奇整数 n，如果 a 和 b 与 n 是互素的，那么

$$\left(\frac{ab}{n}\right)_2 = \left(\frac{a}{n}\right)_2 \cdot \left(\frac{b}{n}\right)_2$$

证明 设 n 的素数分解为

$$n = p_1^{e_1} \cdots p_k^{e_k}$$

那么，根据扩展二次符号的定义

$$\left(\frac{a}{n}\right)_2 = \left(\frac{a}{p_1}\right)_2^{e_1} \cdots \left(\frac{a}{p_k}\right)_2^{e_k}$$

由此，根据前一个推论

$$\left(\frac{a}{n}\right)_2 \cdot \left(\frac{a}{n}\right)_2 = \left(\frac{a}{p_1}\right)_2^{e_1} \cdots \left(\frac{a}{p_k}\right)_2^{e_k} \cdot \left(\frac{b}{p_1}\right)_2^{e_1} \cdots \left(\frac{b}{p_k}\right)_2^{e_k} = \left(\frac{ab}{p_1}\right)_2^{e_1} \cdots \left(\frac{ab}{p_k}\right)_2^{e_k}$$

于是可得

$$\left(\frac{ab}{p_1}\right)_2^{e_1} \cdots \left(\frac{ab}{p_k}\right)_2^{e_k} = \left(\frac{ab}{n}\right)_2$$

这就证明了推论。 ♣

15.4 二次互反律

现在我们介绍**二次互反律**的定理。这是现代数论的第一个结论，其原理早在十八世纪后期就被猜想，但直到1796年才由高斯证明。从直觉来看，绝对没有理由认为这竟然是个正确的结论。但在200年后的今天，这个结论已经被很好地接受和理解并且成为整个*互反律*家族中最具代表性的一个，而这些定律是所谓的“*类域理论*”的一部分，“类域理论”又被所谓的*朗兰兹程序*（Langlands program）所吸收。

在定理的第一部分提到的“互反”这个词，其含义是指如果 p 是 mod q 的一个平方且 q 是 mod p 的一个平方，则它们是*互反*的。定理的证明将在稍后给出。

定理（高斯二次互反） 如果 p,q 是不同的奇素数，那么

$$\left(\frac{p}{q}\right)_2 = (-1)^{(p-1)(q-1)/4}\left(\frac{q}{p}\right)_2$$

在这个公式中出现的表达式 $(-1)^{(p-1)(q-1)/4}$ 仅依赖于 p 和 q 模 4。而且

$$\left(\frac{-1}{p}\right)_2 = (-1)^{(p-1)/2}$$

$$\left(\frac{2}{p}\right)_2 = (-1)^{(p^2-1)/8}$$

特别的，$\left(\frac{-1}{p}\right)_2$ 仅依赖于 p 模 4，$\left(\frac{2}{p}\right)_2$ 仅依赖于 p 模 8。

注意到这个结果是针对勒让德符号的，而不适用于更广泛的雅可比符号。

因此，根据这个结果，我们就有一个算法来判断 $x^2 = c \bmod p$ 是否有解，尽管它不能找出存在的二次根。

备注 证明二次根的存在性和找出存在的二次根之间的区别，完全类似于证明一个数不是素数和找出这个数的一个真因子之间的区别。

例如，我们来看

$$x^2 \equiv 19 \bmod 101$$

是否有解。根据二次互反，我们有

$$\left(\frac{19}{101}\right)_2 = (-1)^{(19-1)(101-1)/4}\left(\frac{101}{19}\right)_2$$

通过考查19和101模4，我们容易发现-1的方幂为1，所以上式左边即可化简为：

$$\left(\frac{101}{19}\right)_2 = \left(\frac{6}{19}\right)_2 = \left(\frac{2}{19}\right)_2 \cdot \left(\frac{3}{19}\right)_2$$

这里用到了前面提到的欧拉定理的推论以及101%19=6。继续分别求解最后两个二次符号的值。对前一个，根据二次互反可得

$$\left(\frac{2}{19}\right)_2 = (-1)^{(19^2-1)/8} = (-1)^{(20\cdot 18)/8} = (-1)^{5\cdot 9} = -1$$

再次应用二次互反定律，通过简约19mod3，另一个二次符号是

$$\left(\frac{3}{19}\right)_2 = (-1)^{(3-1)(19-1)/4}\left(\frac{19}{3}\right)_2 = -\left(\frac{19}{3}\right)_2 = -\left(\frac{1}{3}\right)_2 = -1$$

所以，结果为

$$\left(\frac{19}{101}\right)_2 = \left(\frac{6}{19}\right)_2 = \left(\frac{2}{19}\right)_2 \cdot \left(\frac{3}{19}\right)_2 = (-1)(-1) = 1$$

所以，19是模101的平方。

但实际上，除了去掉2的方幂，没有理由将二次符号中的输入分解为素数。这对大整数尤为重要。

推论（雅可比符号的二次互反） 设 m,n 是正的奇素数，那么，比勒让德符号更常用的是扩展雅可比符号，我们有

$$\left(\frac{m}{n}\right)_2 = (-1)^{(m-1)(n-1)/4}\left(\frac{n}{m}\right)_2$$

证明　当然，我们将应用关于二次互反的高斯定理来证明这个推论。设

$$m = p_1^{e_1}\cdots p_k^{e_k}$$
$$n = q_1^{f_1}\cdots q_\ell^{f_\ell}$$

是 m 和 n 的因式分解。那么雅可比符号 $\left(\frac{m}{n}\right)_2$ 可以按更基本的勒让德符号来表示

$$\left(\frac{m}{n}\right)_2 = \prod_j \left(\frac{m}{q_j}\right)_2^{f_j}$$

根据前面欧拉定理的推论，我们有

$$\left(\frac{m}{q_j}\right)_2 = \prod_i \left(\frac{p_i}{q_j}\right)_2^{e_i}$$

所以，将这两个等式结合起来，我们有

$$\left(\frac{m}{n}\right)_2 = \prod_{i,j} \left(\frac{p_i}{q_j}\right)_2^{e_i f_j}$$

现在，我们将二次互反定律应用在每个 $\left(\frac{p_i}{q_j}\right)_2$ 上我们将得到如下式子：

$$\left(\frac{m}{n}\right)_2 = \prod_{i,j}(-1)^{e_i f_j (p_i-1)(q_j-1)/4}\left(\frac{q_j}{p_i}\right)_2^{e_i f_j}$$
$$= \left(\prod_{i,j}(-1)^{(p_i-1)(q_j-1)/4}\right)\left(\frac{n}{m}\right)_2$$

为了证明这个推论，我们必须证明

$$\prod_{i,j}(-1)^{e_i f_j (p_i-1)(q_j-1)/4} = (-1)^{(m-1)(n-1)/4}$$

由于我们正在讨论 -1 的方幂，所以我们真正需要证明的就是

$$\sum_{i,j}\frac{(p_i-1)(q_j-1)}{4}e_i f_j \equiv \frac{(m-1)(n-1)}{4} \bmod 2$$

去掉分母 4，上式等价于

$$\sum_{i,j}(p_i-1)(q_j-1)e_i f_j \equiv (m-1)(n-1) \bmod 8$$

乐观地看，这只需要我们证明对任意奇整数 a, b, c，有如下等式

$$(ab-1)(c-1) \equiv (a-1)(c-1)+(b-1)(c-1) \bmod 8$$

用三个奇数取代 a, b, c，我们可以使用 $2a+1$，$2b+1$ 和 $2c+1$ 来表示这三个奇数。那么我们希望得到的是

$$((2a+1)(2b+1)-1)((2c+1)-1)$$
$$\equiv((2a+1)-1)((2c+1)-1)+((2b+1)-1)((2c+1)-1) \bmod 8$$

展开这个式子并消去某些项，可以得到

$$(4ab-2a-2b)(2c)=(4ac)+(4bc) \bmod 8$$

去掉公因子4，最后的同余式等价于

$$2abc-ac-bc \equiv ac+bc \bmod 2$$

将所有项都移到左边，这等价于

$$2abc-2ac-2bc \equiv 0 \bmod 2$$

这肯定是正确的。因为这个结论等价于前面对推论的断言，所以我们已经证明了这个推论。（调用二次互反定理） ♣

上述推论的成功证明，扩展了二次互反应用于雅可比符号这一重要情况，我们也需要得到$\left(\frac{-1}{n}\right)_2$和$\left(\frac{2}{n}\right)_2$的相应结果。

推论 设n是正的奇数，我们可以求得雅可比符号的值

$$\left(\frac{-1}{n}\right)_2=(-1)^{(n-1)/2}$$
$$\left(\frac{2}{n}\right)_2=(-1)^{(n^2-1)/8}$$

特别的，$\left(\frac{-1}{n}\right)_2$只依赖于$n$模4，$\left(\frac{2}{n}\right)_2$只依赖于$n$模8。

证明 设$n=p_1^{e_1}\cdots p_k^{e_k}$是$n$的素因子分解，根据用勒让德符号定义的雅可比符号，我们有

$$\left(\frac{-1}{n}\right)_2=\prod_i\left(\frac{-1}{p_i}\right)_2^{e_i}$$

我们知道这个式子的值为

$$\prod_i(-1)^{e_i(p-1)/2}$$

像前面推论中的证明一样，为了证明它等于

$$(-1)^{(n-1)/2}$$

只需要证明对奇数a和b，

$$\frac{a-1}{2}+\frac{b-1}{2}\equiv\frac{ab-1}{2} \bmod 2$$

用$2a+1$和$2b+1$分别代替a和b，并且左右同乘以分母2，这等价于证明

$$(2a+1)-1+(2b+1)-1=(2a+1)(2b+1)-1 \bmod 4$$

将上式展开，则它等价于

$$2a+2b \equiv 4ab+2a+2b \bmod 4$$

这显然是成立的。这证明了关于$\left(\frac{-1}{n}\right)_2$的结论。

对$\left(\frac{2}{n}\right)_2$的情况，如果我们参照上面的步骤，讨论是相同的。雅可比符号的定义有

$$\left(\frac{2}{n}\right)_2=\prod_i\left(\frac{2}{p_i}\right)_2^{e_i}$$

经过二次互反得到

$$\prod_i(-1)^{e_i(p_i^2-1)/8}$$

像上面一样，为了证明它等于

$$(-1)^{(n^2-1)/8}$$

只需要证明对两个任意的奇数a,b，如下更一般的结论成立：

$$(-1)^{((ab)^2-1)/8}=(-1)^{(a^2-1)/8}(-1)^{(b^2-1)/8}$$

像上面一样，为了证明这个关于-1的方幂的结论等价于证明对奇数a,b

$$\frac{(ab)^2-1}{8}\equiv\frac{a^2-1}{8}\frac{b^2-1}{8}\bmod 2$$

左右同乘以分母8并且用$2a+1$和$2b+1$分别代替a和b，这等价于

$$[(2a+1)(2b+1)]^2-1\equiv(2a+1)^2-1+(2b+1)^2-1\bmod 16$$

化简得到

$$(4a^2+4a+1)(4b^2+4b+1)-1\equiv 4a^2+4a+4b^2+4b\bmod 16$$

或者

$$16a^2b^2+16a^2b+16ab^2+16ab+4a^2+4b^2+4a+4b\equiv 4a^2+4a+4b^2+4b\bmod 16$$

如果我们消去16的倍数和左右两边同时出现的项，就会得到等价的同余式$0\equiv 0\bmod 16$。这当然是成立的。

最后，我们对n模4或8的值来验证二次符号$\left(\frac{-1}{n}\right)_2$和$\left(\frac{2}{n}\right)_2$的值。这是简单易行的，对正整数$\ell$，

$$\left(\frac{-1}{n+4\ell}\right)_2=(-1)^{(n+4\ell-1)/2}=(-1)^{2\ell}(-1)^{(n-1)/2}=(-1)^{(n-1)/2}=\left(\frac{-1}{n}\right)_2$$

对$\left(\frac{2}{n}\right)_2$，我们计算

$$\left(\frac{2}{n+8\ell}\right)_2=(-1)^{((n+8\ell)^2-1)/8}=(-1)^{8\ell^2+2\ell n}(-1)^{(n^2-1)/8}$$

这就证明了结论。 ♣

15.5 快速计算

与欧几里得算法有效性的原因类似，二次互反的扩展形式的雅可比符号给出了一个计算二次符号的算法。注意利用雅可比符号，我们就可以避免将整数分解为素数乘积，后者是一

个非常耗时间的事情。

为了具体说明这种方法，我们这里给出两个例子。第一个例子相对要简单一些，我们将给出详细的计算步骤。在第二个例子中，由于使用了大的整数，为了表明算法的确是有效运行的，我们可能略去一些步骤。最后我们得到了该算法更加形式化的描述。

不用进行任何的奇素数因子的分解，我们来计算$\left(\frac{1237}{4327}\right)_2$（整数4327已经是一个素数）。由二次互反律我们有

$$\left(\frac{1237}{4327}\right)_2=(-1)^{\frac{(1237-1)(4327-1)}{4}}\left(\frac{4327}{1237}\right)_2=\left(\frac{4327}{1237}\right)_2$$

注意到我们既不需要知道也不必关心1237是不是素数，用模1237去约简4327，上面的二次符号则等于

$$\left(\frac{616}{1237}\right)_2$$

从616中提取出因子2，则我们得到

$$\left(\frac{2}{1237}\right)_2\left(\frac{308}{1237}\right)_2=\left(\frac{2}{1237}\right)_2\left(\frac{2}{1237}\right)_2\left(\frac{154}{1237}\right)_2$$
$$=\left(\frac{2}{1237}\right)_2\left(\frac{2}{1237}\right)_2\left(\frac{2}{1237}\right)_2\left(\frac{77}{1237}\right)_2$$
$$=\left(\frac{2}{1237}\right)_2\left(\frac{77}{1237}\right)_2$$

这是因为对互素的两个整数m,n而言，总有$\left(\frac{m}{n}\right)_2^2=1$。为了计算$\left(\frac{2}{1237}\right)_2$，我们用模8来约简1237，于是有

$$\left(\frac{2}{1237}\right)_2=\left(\frac{2}{5}\right)_2=(-1)^{(5^2-1)/8}=(-1)^3=-1$$

接下来我们再计算

$$\left(\frac{77}{1237}\right)_2=(-1)^{(77-1)(1237-1)/4}\left(\frac{1237}{77}\right)_2=\left(\frac{1237}{77}\right)_2=\left(\frac{5}{77}\right)_2$$
$$=(-1)^{(77-1)(5-1)/4}\left(\frac{77}{5}\right)_2=\left(\frac{77}{5}\right)_2=\left(\frac{2}{5}\right)_2$$

利用前面的计算结果，

$$\left(\frac{2}{5}\right)_2=(-1)^{(5^2-1)/8}=(-1)^3=-1$$

将两步计算的结果汇总，则有

$$\left(\frac{1237}{4327}\right)_2=\left(\frac{4327}{1237}\right)_2=\left(\frac{616}{1237}\right)_2=\left(\frac{2}{1237}\right)_2\left(\frac{77}{1237}\right)_2$$
$$=(-1)\left(\frac{1237}{77}\right)_2=(-1)\left(\frac{5}{77}\right)_2=(-1)\left(\frac{77}{5}\right)_2=(-1)\left(\frac{2}{5}\right)_2=(-1)(-1)=1$$

因此，我们说 1237 是模 4327 的一个平方根。

现在让我们来给出一个大点的数的例子，计算

$$\left(\frac{123456791}{987654323}\right)_2$$

这里我们已知 987654323 是一个素数。利用二次互反律，并且注意到这两整数模 4 后均约简为 2，因此

$$\left(\frac{123456791}{987654323}\right)_2=-\left(\frac{987654323}{123456791}\right)_2=-\left(\frac{123456786}{123456791}\right)_2$$

（这里就需要一些计算的技巧，略去一些计算步骤）提取出 2 的方幂，可得

$$-\left(\frac{2}{123456791}\right)_2\left(\frac{61728393}{123456791}\right)_2=-\left(\frac{2}{7}\right)_2\left(\frac{61728393}{123456791}\right)_2=-\left(\frac{61728393}{123456791}\right)_2$$

因为 61728393%4 = 1，因此

$$-\left(\frac{123456791}{61728393}\right)_2=-\left(\frac{5}{61728393}\right)_2=-\left(\frac{61728393}{5}\right)_2=-\left(\frac{3}{5}\right)_2$$

$$=-\left(\frac{5}{3}\right)_2=-\left(\frac{2}{3}\right)_2=-(-1)=1$$

因此，123456791 是模 987654323 的一个平方。

在上述第二个例子中，我们的运气还算不错，算法很快就结束了。一般地，类似于欧几里得算法中的大概估计，计算 $\left(\frac{x}{n}\right)_2$ 最多需要 $2\log_2 n$ 步，这里的 x 满足 $1<x<n$。但是，因子 2 的出现还需要额外的处理。

习题

15.5.01 计算二次符号 $(-1/1009)_2$ 的值，判断 −1 是不是模 1009 的一个平方？

15.5.02 计算二次符号 $(-1/1033)_2$ 的值，判断 −1 是不是模 1033 的一个平方？

15.5.03 计算二次符号 $(2/1009)_2$ 的值，判断 2 是不是模 1009 的一个平方？

15.5.04 计算二次符号 $(2/1033)_2$ 的值，判断 2 是不是模 1033 的一个平方？

15.5.05 计算二次符号 $(3/1009)_2$ 的值，判断 3 是不是模 1009 的一个平方？

15.5.06 计算二次符号 $(-5/1009)_2$ 的值，判断 −5 是不是模 1009 的一个平方？

15.5.07 计算二次符号 $(119/1009)_2$ 的值，判断 119 是不是模 1009 的一个平方？

15.5.08 计算二次符号 $(-1/2009)_2$ 的值，判断 −1 是不是模 2009 的一个平方？

15.5.09 计算二次符号 $(-1/2033)_2$ 的值，判断 −1 是不是模 2033 的一个平方？

15.5.10 计算二次符号 $(-1/2041)_2$ 的值，判断 −1 是不是模 2041 的一个平方？

15.5.11 计算二次符号 $(2/2009)_2$ 的值，判断 2 是不是模 2009 的一个平方？

15.5.12 计算二次符号 $(3/2009)_2$ 的值，判断 3 是不是模 2009 的一个平方？

15.5.13 计算二次符号 $(-5/2009)_2$ 的值，判断 −5 是不是模 2009 的一个平方？

15.5.14 计算二次符号 $(119/2009)_2$ 的值，判断 119 是不是模 2009 的一个平方？

15.5.15(*) 利用欧拉准则与利用二次互反律，哪一个方法在计算模素数 p 的平方根 x 时更快或者更好？

第16章 伪 素 数

我们所知道最简单的素性判别的方法是**试除法**，这大概需要 $\sqrt{n}$ 步才能证明 n 是素数。当 $n \approx 10^{18}$ 时，在主频为 200MHz 的计算机上进行验证大概需要几分钟的时间，但当 $n \approx 10^{60}$ 时，这个时间则变为 10^{16} 年。然而现代密码系统所需要的许多素数至少是要大于 10^{60}。

首先要折衷的一点是，如果我们牺牲一些确定性就可以在速度上获得大的提升，也就是说，我们可以快速的证明很大的数可能是素数，但这样并不能像传统计算方法那样得到非常确定的素数。由于这些传统的计算方法从来就没有被完成，所以某种程度的“牺牲”这个说法也许不大合适。

我们称已经通过不同的素性概率检验，但还不能确定是否真正是素数的数为**伪素数**（有不同类型）或**可能素数**。有时“**伪素数**”这个词通常是指一个通过了素性概率检验的非素数。然而对我们而言，一个伪素数是指通过了一系列素性概率检验的数（可能是素数，也可能不是）。

对数 n 进行某个特定的素性概率检验都需要利用一个或多个辅助的数 b，这个 b 是从 $1<b<n$ “随机”选取的。如果一个特定的辅助数 b 告诉我们“n 可能是素数”，则 b 是 n 的**素性证据**。问题是在 1 到 n 之间绝大部分数都**不是证据**（有时称之为**假证据**），也就是说它们的结论是 n 是素数，而实际上 n 并不是素数。因此，这个问题的一部分就是要保证在 1 到 n 之间选取的数 b 是判断 n 是素数或是合数的（真的）证据。费马伪素数检验的致命缺陷就是存在没有证据的合数 n，这称为**卡米克尔数**。其他两个素性检验则没有这个缺陷。

在所有的情况下，这个用在我们所说“n 是素数的概率是 2^{-10}”的概率概念是一个基础的启发式概念，它基于这样一个假设，在 n 种可能性中我们不能理解每个可能性都以 $1/n$ 的概率发生。（这种伪概率推理在过去的 200 年一直不为重视）

另一方面，如果扩展黎曼假设是真的，这些素性概率检验可以转化为确定性检验。许多数学家相信**扩展黎曼假设**是真的，而且没有能反驳这一点的简单证明（直到 2000 年中期），但似乎也还没有人知道怎么证明这一点。这个问题的提出已经有 140 年，但一直没有实质的进展。假定扩展黎曼假设是真的，就可以得出存在一个惟一的常数 C，对任意的整数 n，如果 n 是合数，则存在一个欧拉证据（这也是个强证据）b

$$1<b<C\cdot(\log n)^2$$

也就是说，如果 n 是合数，我们可以缩小查找一个（真）证据的范围。但具有讽刺意味的是，即使是这个结论的精练形式，甚至接受扩展黎曼假设，要确定大数的素数性还是需要进行几百或几千次概率算法所需的米勒-罗宾检验。也就是说，即使我们有一个确定的方法可用，概率方法还是显得更快一些。这使得我们想知道需要付出多少才能得到完全的确定的素性，而不是 99.9999999% 的概率素性。

16.1 费马伪素数

这一节给出了一个素性的启发式检验。它有几个缺陷而且最终我们不会用到它，但它说

明了两个非常重要的问题：首先，概率算法要比确定性算法更快；其次，我们不能简单的期望能够对所有看上去是真实的事情提供证据。

一方面，费马所谓的小定理，断言对任何素数 p 和整数 b 有

$$b^p = b \bmod p$$

等价的描述则是，如果 p 不整除 b，我们有

$$b^{p-1} = 1 \bmod p$$

还有欧拉定理的一个特例，它断言如果 b 和整数 n 互素，则

$$b^{\varphi(n)} = 1 \bmod n$$

其中 φ 是欧拉函数。（应用群论可以很好的证明欧拉定理，我们将在稍后做这件事情。）

整数 n 被称为**费马伪素数**或**普通伪素数**，或者简单地称为**伪素数**，如果

$$2^{n-1} = 1 \bmod n$$

因为费马小定理的逆命题不成立，所以说 n 是一个费马伪素数并不保证它就是一个素数。而且在实际中满足 $2^n = 2 \bmod n$，而 n 不是素数这种情况并不少见。比如，$341 = 11 \times 31$ 不是素数，而且这是最小的是费马伪素数而不是素数的数：$2^{341} = 2 \bmod 341$。其余的一些非素费马伪素数是：

561 645 1105 1387 1729 1905 2047 2465

在 10^9 以内只有 5597 个非素的费马伪素数。

如果一个整数 n 没有通过费马检验，也就是 $2^{n-1} \neq 1 \bmod n$，则 n 一定不是素数（由于如果 n 是素数，则必然有 $2^{n-1} = 1 \bmod n$）。

因为快速指数算法为计算 $b^{n-1} \bmod n$ 提供了一个经济简便的方法，所以我们可以比用试除法验证一个整数 n 的素数性更快地检验其是否为费马伪素数。

我们可以给一个更严格的条件：整数 n 对于**基数 b 是费马伪素数**，如果

$$b^{n-1} = 1 \bmod n$$

如果 $b^{n-1} \neq 1 \bmod n$，则 n 一定不是素数。另一方面，即使对所有与 n 互素的整数 b 都有 $b^{n-1} = 1 \bmod n$，我们还是不能保证 n 就是素数。

一个整数 b 满足 $1 < b < n$，如果 $b^{n-1} = 1 \bmod n$ 则说 b 是 n 的素性（**费马**）**证据**，如果 n 实际不是一个素数，则说 b 是 n 的（费马）**非证据**或（费马）**假证据**。

而且一个非素数对不同的基数有不同的表现。例如，非素数 $91 = 7 \cdot 13$ 不是费马伪素数（基数是 2），但如果基数是 3 的话就是费马伪素数。

在实际中，一个整数对于一个基数或更多的基数 b 是伪素数而它并不是素数的情况是常见的。而且有无穷多个整数对所有的基数（和它们互素的数）都是伪素数，但实际却不是素数。这称为卡米克尔数，小于 10 000 的**卡米克尔数**有

561 1105 1729 2465 2821 6601 8911

在 10^9 以内只有 2163 个卡米克尔数，在 10^{12} 以内有 8241 个，在 10^{13} 以内有 19 279 个，在 10^{14} 以内有 44 706 个，在 10^{15} 以内有 105 212 个（参见[Pinch 1993]）。

稍后，我们将证明一个卡米克尔数一定是奇数、非平方，并且至少可以被 3 个素数整除的结论。这个结论对于理解为什么更优的伪素数和相应概率素性检验方法（下面）没有现有的卡米克尔数类似的缺点也是十分必要的。

是否有无穷多个卡米克尔数是一个已经有 80 年历史的公开问题。最近，有人证明卡米克

尔数是无穷多的：实际上，存在一个常数 C 使得小于 x 的卡米克尔数的数量至少是 $C\cdot x^{2/7}$。参见[Alfold，Granville，Pomerance 1994]。

习题

16.1.01 证明合数 45 对基数 17 是费马伪素数。

16.1.02 证明合数 49 对基数 18 是费马伪素数。

16.1.03 证明合数 51 对基数 35 是费马伪素数。

16.1.04 证明合数 55 对基数 21 是费马伪素数。

16.1.05 证明合数 57 对基数 20 是费马伪素数。

16.1.06 证明合数 65 对基数 14 是费马伪素数。

16.1.07 验证 341,561,645,1105 不是素数，但它们对于基数 2 是最小的费马伪素数。这些非素数中哪个可以通过对于基数 3 或者基数 5 的费马检验？

16.1.08 证明如果 n 对于基数 2 和基数 3 是费马伪素数，则它对于基数 6 也是费马伪素数。

16.2 非素的伪素数

我们可以很容易地证明，对固定的基数 b，如果有一个对基数 b 的非素数费马伪素数，则我们可以明确得到无限多这样的数。实际上，比这更好的是，对基数 b 的无限多非素数费马伪素数存在一个清晰的结构。这个思想由 A.Korselt 在 1899 年、E.Malo 在 1903 年、M.Cipolla 在 1904 年、R.D.Carmichael 在 1912 年分别提出。

第一个命题假定我们已经知道一个对基数 b 是非素的伪素数，由此我们可以生成一个迅速增加的这类整数的无限序列。

命题 假设 $b^{n-1}=1 \bmod n$，但 n 是一个合数并且 $gcd(b-1,n)=1$，那么

$$N=\frac{b^n-1}{b-1}$$

具有同样的性质：即 $b^{N-1}=1 \bmod N$，但 N 是合数而且和 $b-1$ 互素。因此，从一个对基数 b 的非素的费马伪素数 n 出发，我们可以得到另外一个这样的数 N。

证明 首先，

$$N-1=\frac{b^n-1}{b-1}-1=\frac{b^n-1-b+1}{b-1}=b\cdot\frac{b^{n-1}-1}{b-1}$$

由于 n 和 $b-1$ 没有公因子，且由假设 $n\mid b^{n-1}-1$，则 $n\mid\dfrac{b^{n-1}-1}{b-1}$。所以存在整数 ℓ，使

$$\frac{b^{n-1}-1}{b-1}=n\ell$$

也就是说，我们有

$$N-1=b\cdot\frac{b^{n-1}-1}{b-1}=bn\ell$$

那么

$$b^{N-1}-1=b^{bn\ell}-1=(b^n)^{b\ell}-1$$

根据基本的代数知识可知这是 b^n-1 的倍数，所以当然也是 $N=(b^n-1)/(b-1)$ 的倍数。最后，

为了验证 N 和 $b-1$ 是互素的，根据费马观察，对于 $b>1$ 有 $gcd(b^m-1,b^n-1)=b^{gcd(m,m)}-1$，所以有

$$gcd(b-1,b^n-1)=b-1,\ gcd(b-1,(b^n-1)/(b-1))=1$$

证明完成。 ♣

下一个命题将给出对于基数 b 的一个无限多个非素的费马伪素数集合。

命题 设 $b\geq 2$，p 是满足 $p>2$ 且不整除 $b(b-1)(b+1)$ 的素数。则

$$N=\frac{b^p-1}{b-1}\cdot\frac{b^p+1}{b+1}$$

是一个对于基数 b 的非素费马伪素数。

证明 因为 $b\geq 2$ 且 $p>2$ 是奇数，$(b^p-1)/(b-1)$ 和 $(b^p+1)/(b+1)$ 都是大于1的整数，所以 N 肯定不是素数。则

$$\frac{b^p-1}{b-1}\cdot\frac{b^p+1}{b+1}-1=\frac{b^2((b^2)^{p-1}-1)}{b^2-1}$$

由于 p 是素数且不整数 b，所以 p 整除 $(b^2)^{p-1}-1$。因为 p 不整除 b^2-1，所以实际上 p 整除

$$\frac{(b^2)^{p-1}-1}{b^2-1}$$

而且，如果不考虑 b 的奇偶性，那么

$$\frac{b^2((b^2)^{p-1}-1)}{b^2-1}$$

是偶数。所以令

$$\frac{b^2((b^2)^{p-1}-1)}{b^2-1}=2\ell p$$

其中 ℓ 是一个整数。那么

$$b^{\frac{b^p-1}{b-1}\cdot\frac{b^p+1}{b+1}-1}-1=b^{2\ell p}-1$$

由基本的代数知识可知它是 $b^{2p}-1$ 的倍数。因此，可以肯定，它也是 $N=(b^{2p}-1)/(b^2-1)$ 的倍数。所以，N 是对于基数 b 的非素费马伪素数。 ♣

备注 相比较而言，要证明卡米克尔数是无穷多的更困难些，也就是说对非素数 n，要证明所有与 n 互素的 b 都满足 $b^{n-1}=1 \bmod n$。这一点直到1994年才由 Alford 、Granville 和 Pomerance 等人完成。参见[Alford ,Granville,Pomerance 1994]。

习题

16.2.01 找出五个对于基数2的不同非素费马伪素数。

16.2.02 找出五个对于基数3的不同非素费马伪素数。

16.2.03 找出六个对于基数7的不同非素费马伪素数。

16.2.04(*) 找出不同的五个同时是对于基数2和基数3的费马伪素数的非素数。

16.3 欧拉伪素数

可以改进我们的素性概率检验算法，以成功排除某些是（费马）伪素数的非素数。这需

要一些附加的结构。于1976年提出的索洛维-斯特拉森检验是第一个非常合理的综合素性概率检验算法。其合理的快速算法依赖于利用二次互反律对（雅可比）二次符号的快速估算。我们称一个通过索洛维-斯特拉森检验（可能是也可能不是素数）而“可能是素数”的数为**欧拉伪素数**。重要的是注意在欧拉伪素数中没有类似于卡米克尔数的数。

对给定的模数 n 和数 b，问题是 b 是否有一个模 n 的平方根。也就是说，如下方程

$$x^2 = b \bmod n$$

是否有解？

对于奇素数 p，**二次符号或勒让德符号** $(b/p)_2$（有时简单地记作 (b/p)，将下标省略）定义为

$$\left(\frac{b}{p}\right)_2 = \begin{cases} 0 & \text{如果} gcd(b,p) > 1 \\ 1 & \text{如果} x^2 = b \bmod p \text{有解且} gcd(b,p) = 1 \\ -1 & \text{如果} x^2 = b \bmod p \text{无解且} gcd(b,p) = 1 \end{cases}$$

则对任意的整数 n，且其素因子分解为

$$n = 2^{e_0} p_1^{e_1} \cdots p_k^{e_k}$$

其中 p_i 是奇素数，**扩展二次符号或雅可比符号**定义为

$$\left(\frac{b}{n}\right)_2 = \left(\frac{b}{p_1}\right)_2^{e_1} \cdots \left(\frac{b}{p_k}\right)_2^{e_k}$$

重要的是，认识到素数 p 的二次符号 $(b/p)_2$ 已经说明 b 是否是模 p 的平方，但对于非素数 n 就不能说明这一点。也就是说，$(b/n)_2$ 这个值并不能直接说明 $x^2 = b \bmod n$ 是否可解。然而，对于目前的目标而言，雅可比符号的优势在于可以快速的计算。（如果 $(b/n)_2 = -1$，则明确说明 b 不是一个平方；但如果 $(b/n)_2 = 1$，就还难以确定 b 是否是一个平方。）

欧拉证明了（这一点我们将在稍后进行）对素数 p 有

$$\left(\frac{b}{p}\right)_2 = b^{\frac{p-1}{2}} \bmod p$$

相反，如果 n 不是素数，$b^{\frac{n-1}{2}}$ 对于不同的 b 就是随机的。

对于欧拉伪素数采用同样的不合理跳跃，我们认为整数 n 是基为 b 的欧拉伪素数如果有下式

$$\left(\frac{b}{n}\right)_2 = b^{(n-1)/2} \bmod n$$

由于二次符号的值为 ±1 和 0，我们可以看到一个对于基数 b 的欧拉伪素数同时是一个对于基数 b 的费马伪素数。毕竟，我们需要应用费马判别准则来计算的部分是模 n 的平方，而这可以应用欧拉判别准则来计算。

当然，为了让这个准则能在所有的情况下使用，我们必须能快速计算 $b^{(n-1)/2} \bmod n$ 和雅可比符号。快速指数算法能够实现前者，而**二次互反律**则提供了对后者的快速算法。

满足 $b^{(n-1)/2} = \left(\frac{b}{n}\right)_2$ 的整数 b 是 n 的素性的一个**欧拉证据**。如果这个等式成立而 n 不是素数，则 b 是一个**非（欧拉）证据**，或**假证据**。

稍后我们将看到在欧拉伪素数中没有和卡米克尔数类似的数。也就是说，如果n不是素数，则至少有一半的b在$0<b<n$范围内是这个事实的证据。剩下的问题就是我们尚不清楚这些证据是如何分布的。

16.4 索洛维–斯特拉森检验

这个检验是第一个于1976年提出的素性概率检验算法。当对n应用这个检验时，它对于不同的基数b检验n是否是欧拉伪素数。其合理的快速算法依赖于二次互反律允许对（雅可比）二次符号的快速估算。但它的编程实现要比米勒–罗宾检验困难，所以在实际中米勒–罗宾检验用的更多。（另一方面，完全理解米勒–罗宾检验的工作原理还稍有困难。）

需要注意的是，由于米勒–罗宾更简单并且结果更佳，所以索洛维–斯特拉森检验（Soloray–Strassen Test）并不适用于实际的目的。然而，索洛维–斯特拉森的运行机制还是值得研究的，它引出了稍后要用到的重要问题和概念。

索洛维–斯特拉森检验可以证明合数的确定性，但只能以某个启发式概率证明素数性。

我们将描述索洛维–斯特拉森算法的工作过程，在解释它是如何工作之前我们需要更多的准备工作。

设n是正奇数，在1到$n-1$之间“随机地”选取k个整数b，对所选取的每个b计算

$$b^{(n-1)/2}\ \%\ n$$

和（雅可比）二次符号

$$\left(\frac{b}{n}\right)_2$$

如果用所选取的b计算出的这两个数值不相等，就停下来：n不是素数。如果这两个表达式的值对所选的b都相等，则我们设想n是素数的概率至少为

$$1-2^{-k}$$

这个思想是，如果n是合数，则在1到$n-1$之间至少有一半的数是这个事实的证据，也就是说，存在n和b，满足

$$b^{(n-1)/2}\%n \neq \left(\frac{b}{n}\right)_2$$

这样的一个b被称作n的合数性的**欧拉证据**。

对每个b，你可以验证每步计算需要进行$O(\log_2^3 n)$次位运算。

重要的是确信对任何合数n，对于$1<b<n$至少有一半的b是证据。在理解了本原根（和诸如二次互反等相关概念）的概念之后我们可以证明这一点。

的确，要完全理解索洛维–斯特拉森检验的运行机制或概率是怎样得到的并不容易。

习题

16.4.01 找出三个最小的卡米克尔数 561、1105、1729 是合数的最小欧拉（索洛维–斯特拉森）证据。

16.5 强伪素数

继续应用平方根来检验素数性，我们可以比欧拉准则走得更远些。这里的基本思想，是

如果 p 是一个素数，则 $\mathbf{Z}/p$ 中应该只有1的两个平方根，也就是 ± 1。

设 n 是奇数，因式分解

$$n-1=2^s\cdot \ell$$

其中 ℓ 是奇数。则 n 是基数为 b 的**强伪素数**，如果

$$b^{\ell}=1 \bmod n \quad \text{或者} \quad \text{对某些} 0\leqslant r<s, \quad b^{2^r\cdot \ell}=-1 \bmod n$$

表面上看，确实很难看出这和素数性有什么联系。除非上面关于这个检验的备注和1的“错误的”平方根的出现是相关的，否则确实很难理解这是怎样一回事。

无论如何，考虑到快速指数算法，这个算法的速度是相当快的。其工作原理我们稍后将介绍。

16.6 米勒-罗宾检验

米勒-罗宾（Miller-Rabin）素性概率检验算法可以用来寻找强伪素数。当应用在数 n 上时，它用几个不同的基数 b 检验 n 是否是强伪素数。这个检验易于运行，所以在实际中应用广泛。其速度很快的原因是模 n 的指数运算相当快。

这个检验的思想是对非素数 n，在 $\mathbf{Z}/n$ 中至少存在两个元素 x 使得 $x^2=1$ 而且 $x\neq \pm 1$。也就是说，1 有多于两个的平方根。和索洛维-斯特拉森检验一样，对其工作原理做更多的解释是必要的。米勒-罗宾本身的原理相当简单，甚至比索洛维-斯特拉森检验还简单。

和索洛维-斯特拉森检验一样，它可以证明合数的确定性，但只能以某种概率说明素数性。基于这个原因，我们有时将米勒-罗宾作为一个**合数性检验**。

设 n 是正奇数，找出能整除 $n-1$ 的最大方幂 2^r，则令 $n-1=2^r\cdot m$（所以 m 是奇数）为了发现

- n 肯定是合数

或者对 k 个满足 $1<b<n-1$ 的整数 b

- n 是素数的概率 $\geqslant 1-(1/4)^k$

对所选的每个 b，做如下计算：

- 计算 $b_1=b^m \% n$
- 如果 $b_1=\pm 1 \bmod n$，停止：这个 b 是 n 以 $\geqslant 3/4$ 的概率是素数的证据。
- 否则继续：对 $s<r$，计算 $b_s=(b_{s-1})^2 \% n$。如果 $b_s=-1 \bmod n$，停止：n 以 $\geqslant 3/4$ 的概率是素数。$b_s=1 \bmod n$（$s>1$），整个过程终止：n 肯定是合数。
- 如果 $b_2,b_3,\cdots,b_{r-1}$ 中都没有一个是 -1，整个过程终止：n 肯定是合数。
- 如果所选的每个 b 都说明 n 以 $\geqslant 3/4$ 的概率是素数，则我们设想 n 是素数的概率为 $1-(1/4)^k$，根据假设这个检验中相应 b 的选取是相互独立的。

这里的主要思想（也就是概率的根据）就是如果 n 是合数，则在 $1<b<n-1$ 范围中至少有 $3/4$ 的 b 是这个事实的证据。和索洛维-斯特拉森检验一样，证明有这么多的证据需要进行其他准备，所以我们把这件事留待后面讨论。

习题

16.6.01 证明合数1281对基数41是强伪素数。

16.6.02 证明合数1729对基数10是强伪素数。

16.6.03 证明合数3073对基数1146是强伪素数。

16.6.04 证明合数3201对基数1163是强伪素数。

16.6.05 证明合数3585对基数434是强伪素数。

16.6.06 证明合数5377对基数848是强伪素数。

16.6.07 找出卡米克尔数2465、2821是合数的最小强（米勒-罗宾）证据。

第17章 群

这里我们碰到了抽象代数中的第一个例子，而不是高中所学过的有形代数的例子。这样考虑问题主要是因为这是尝试直接研究事物结构的方法，而不去参考不相关的特定细节。

从长远观点看，这可以达到令人称奇的效果，因为它将表明类似这样的结构在数学中反复出现。因此，仔细研究这些基本的结构，可以使我们掌握不同现象内在更简单并且更统一的本质。

17.1 群概念

抽象代数中最简单的（但可能不是直觉上最直接的）研究对象就是群。这个概念已深入现代数学了。许多看似初等的问题仅仅是有关群基本事实的神秘表现。这一点在初等数论中表现得尤为明显，许多结论都可以给出“初等的”证明，但这样的证明却需要掌握复杂并且容易混淆的知识。

群 G 是具有被称为**单位**的特殊元素、定义了运算 $g*h$ 并且满足如下性质的集合：

- 存在单位元素：即对所有的 $g\in G$，$e*g=g*e=g$；
- 存在**逆元素**：即对所有的 $g\in G$，存在 $h\in G$（g 的**逆元素**）满足 $g*h=h*g=e$；
- 结合律：对所有的 $x,y,z\in G$，$x*(y*z)=(x*y)*z$。

如果运算 $g*h$ 是可交换的，即如果 $g*h=h*g$，则称这样的群是**阿贝尔群**（因数学家 N.H.Abel 而得名）。这时，通常将运算记为加法（但不总是这样）。如果运算记为加法，则群的单位 e 记为 0。

在许多情况下，群的运算还记为乘法的形式，如 $g*h=g\cdot h=gh$。这也不排除群的运算是可交换的情况，只是需要注意这里并没有假设群就是阿贝尔群。如果群的运算记为乘法，则其单位 e 就记为 1。通常当群的运算简单地记为乘法时，群的元素 g 的**逆元素**就表示为

$$g\text{ 的逆元素}=g^{-1}$$

如果群的运算记为加法时，群的元素 g 的逆元素就表示为

$$g\text{ 的逆元素}=-g$$

除非我们能够证明群的每一个元素确实存在一个逆元素，否则这种记法可能是不合适的。

在下面的每一个例子中，容易验证它们满足群的几个必要性质，我们只需要验证它们的单位和逆元即可。

- 整数集合 $\mathbf{Z}$，运算为通常的加法+。单位元为 0，任一元素 x 的逆元素为 $-x$。该群为阿贝尔群。
- 偶整数集合 $2\mathbf{Z}$，运算为通常的加法+。单位元为 0，任一元素 x 的逆元素为 $-x$。该群为阿贝尔群。
- 整数中 7 的倍数组成的集合 $7\mathbf{Z}$，运算为通常的加法+。单位元为 0，任一元素 x 的逆元素为 $-x$。该群为阿贝尔群。

- 整数模 m 的集合 $\mathbf{Z}/m$，运算定义为模 m 加法。单位元为 0 模 m，任一元素 x 的逆元素为 $(-x)$ 模 m。该群为阿贝尔群。
- 整数模 m 且与 m 互素的整数集合 $\mathbf{Z}/m^{\times}$，运算定义为模 m 乘法。单位元为 1 模 m。在这个例子中，不熟悉模 m 运算的人可能认识不到存在乘法逆元素。实际上我们可以通过欧几里得算法计算出来。所以这是首个非平凡的例子。该群是阿贝尔群。
- n 维实数空间 $\mathbf{R}^n$ 中向量的集合，运算定义为向量加法。单位元则为 0 向量，逆元素为向量的负向量（注意，这里我们忽略了存在标量乘法这一事实）。
- 2×2 可逆实矩阵的集合 $GL(2,\mathbf{R})$，运算定义为矩阵乘法。单位元为 2×2 单位矩阵 $\begin{pmatrix}1&0\\0&1\end{pmatrix}$，逆元素的存在性是显然的。矩阵乘法的结合律从定义可能不是很明显，但可用手工检验或者由更深的原理推导出来。两个可逆矩阵的乘积是可逆的这一事实是有趣的：假设 g,h 均有相应的逆矩阵 g^{-1} 和 h^{-1}，则 $h^{-1}g^{-1}$ 是 gh 的逆矩阵。这个群当然不是阿贝尔群。
- 置换的集合构成群，运算定义为置换的合成。单位置换就是该群的单位元，因为置换是映射，结合律自然满足。如果该置换集合的元素多于两个，则该置换群当然就是非阿贝尔群。
- 集合 S 到自身的双射函数的集合构成一个群，运算定义为函数的合成。单位元就是函数 e，它将每一个元素映射到自身，即对所有的 $s\in S$，$e(s)=s$（注意，这个例子仅仅是比前面置换群的例子更一般的情况）。

习题

17.1.01 证明在任意的群 G 中，对任意元素 $h,x,y\in G$，都有

$$h(xy)h^{-1}=(hxh^{-1})(hyh^{-1})$$

17.1.02 用归纳法证明在任意的群 G 中，对任意的元素 $g,h\in G$ 和任意的整数 n，下式成立：$hg^nh^{-1}=(hgh^{-1})^n$。

17.1.03 构造 $\mathbf{Z}/4$ 的加法表和 $\mathbf{Z}/5^{\times}$ 的乘法表。

17.1.04 为什么集合 $\{1,2,3,4,5\}$ 加上模 6 乘法运算不构成群？

17.1.05 用归纳法证明在任意的阿贝尔群 G 中，对所有元素 $g,h\in G$ 和所有的正整数 n，都有 $(gh)^n=g^nh^n$。

17.1.06 证明在某个群中当且仅当 $gh=hg$，$(gh)^2=g^2h^2$。

17.1.07 证明 $(gh)^{-1}=h^{-1}g^{-1}$。

17.1.08 证明当且仅当 $gh=hg$，$(gh)^{-1}=g^{-1}h^{-1}$。

17.2 子群

子群是群的一个子集，它保持了群的运算性质。对于群 G 的一个子集 H，如果在群 G 的运算下它是一个群，则称 H 是 G 的一个子群。

这也就是说，如果 H 包含单位元素 $e\in G$，如果 H 包含它所有元素的逆元素，并且如果 H 包含它任意两个元素的乘积，则 H 就是一个子群。（运算的结合律在这里是有保证的，因为 G 本身是一个群，它的运算是满足结合律的。）

另一个解释则是：如果$e\in H$，并且如果对所有的$h\in H$，其逆元素h^{-1}也属于H，此外对所有的$h_1,h_2\in H$，h_1h_2也属于H，则H是一个子群。

另一个更简洁的解释是：如果$e\in H$，并且对所有的$h_1,h_2\in H$，$h_1h_2^{-1}$也属于H，则H是G的一个子群。（如果我们取$h_1=e$，则后面的条件就确保了逆元素的存在。）

此时，我们通常可以认为**H对求逆是封闭的**并且**对群的运算是封闭的。**

例如，所有偶整数集合是整数加法群的一个子群。更一般地，对于给定的整数m，所有m的倍数组成的集合H就是整数加法群的一个子群。我们来验证这一结论，首先单位元0是m的倍数，所以$0\in H$。对任意两个被m整除的整数x,y，可以记$x=ma,y=mb$，a和b是整数，则根据上述比较简洁的解释，我们有

$$x-y=ma-mb=m(a-b)\in H$$

故H对求逆是封闭的并且对群的运算是封闭的。因此它也是$\mathbf{Z}$的一个子群。

习题

17.2.01 验证$H=\{0-\text{mod}-8,\ 2-\text{mod}-8,\ 4-\text{mod}-8,\ 6-\text{mod}-8\}$是$\mathbf{Z}/8$加法群的一个子群。

17.2.02 验证2的方幂模7的集合是$\mathbf{Z}/7^{\times}$的一个子群。

17.2.03 证明群G的两个子群H和K的交集$H\cap K$仍是G的一个子群。

17.2.04 证明在一个阿贝尔群G中，对于一个给定的整数n，集合$X_n=\{g\mid g\in G,g^n=e\}$是$G$的一个子群。

17.2.05 求$\mathbf{Z}/16$（加法）的所有5个不同的子群。

17.2.06 求$\mathbf{Z}/24$（加法）的所有8个不同的子群。

17.2.07 群$\mathbf{Z}/30^{\times}$有8个子群，请将它们全部找出来。

17.2.08 验证$GL(2,\mathbf{Q})$中形如$g=\begin{pmatrix}a&0\\0&d\end{pmatrix}$（即左下角和右上角元素为0）的矩阵组成的集合是$GL(2,\mathbf{Q})$的一个子群。

17.2.09 验证$GL(2,\mathbf{Q})$中形如$g=\begin{pmatrix}a&b\\0&d\end{pmatrix}$（即左下角元素为0）的矩阵组成的集合是$GL(2,\mathbf{Q})$的一个子群。

17.3 拉格朗日定理

本节的定理是群论作为结构计数（structured counting）应用的最简单的例子。尽管本节的讨论完全是抽象的，但它将给出欧拉定理的最简单的证明方法。**有限群**就是包含有限个元素的群。有限群的**阶**就是该群所含元素的个数。有时我们用$|G|$表示群G的阶。在本节中，我们就将群的运算简单地记为乘法。

定理（拉格朗日） G是一个有限群，H是G的一个子群，则H的阶整除G的阶。

为了证明这个定理，我们需要了解一些概念，这几个概念在后面还会用到。对群G的一个子群H，$g\in G$，则由g所形成的H的**左陪集**或**左平移**为

$$gH=\{gh:h\in H\}$$

类似地，由g所形成的H的**右陪集**或**右平移**为

$$Hg = \{hg : h \in H\}$$

证明 首先我们将证明 H 的所有左陪集的集合是 G 的一个划分，这也意味着 G 的每一个元素都存在于 H 的某个左陪集中，并且如果两个左陪集 xH 和 yH 的交不为空，则必有 $xH = yH$。（注意这并不意味着 $x = y$）

因为 $x = x.e \in xH$，所以 G 的每个元素都存在于 H 的某个左陪集中。

现在我们假设对 $x, y \in G$，$xH \cap yH \neq \varnothing$，那么就存在 $h_1, h_2 \in H$ 使得 $xh_1 = yh_2$。给这个等式两边同时右乘 h_2^{-1} 得到

$$(xh_1)h_2^{-1} = (yh_2)h_2^{-1}$$

上述等式的右边可以化简为

$$\begin{aligned}(yh_2)h_2^{-1} &= y(h_2h_2^{-1}) && \text{（由结合律）}\\ &= y.e && \text{（由逆元性质）}\\ &= y && \text{（由单位元性质）}\end{aligned}$$

为了描述简便，令 $z = h_1h_2^{-1}$，由 G 的结合律

$$y = (xh_1)h_2^{-1} = x(h_1h_2^{-1}) = xz$$

因为 H 是一个子群，$z \in H$，则

$$yH = \{yh : h \in H\} = \{(xz)h : h \in H\} = \{x(zh) : h \in H\}$$

一方面，因为 H 对乘法是封闭的，所以对于每一 $h \in H$，乘积 zh 仍含于 H 中。因此

$$yH = \{x(zh) : h \in H\} \subset \{xh' : h' \in H\} = xH$$

尽而有 $yH \subset xH$。但由于 x 和 y 之间的关系完全是对称的，因此有 $xH \subset yH$，于是有 $xH=yH$。（换句话说，我们已经证明 G 中 H 的左陪集真正实现了对 G 的划分）

接下来我们将证明 H 的左陪集的基数都是相同的。为了得到这一结论，我们将证明对任意的 $x \in G$，存在一个 H 到 xH 的双射。特别地，定义

$$f(g) = xg$$

（很显然这个映射将 H 映射到了 xH）其次，我们将证明单射性：如果 $f(g) = f(g')$，即

$$xg = xg'$$

两边左乘以 x^{-1} 则得到

$$x^{-1}(xg) = x^{-1}(xg')$$

运用结合律得到

$$(x^{-1}x)g = (x^{-1}x)g'$$

再利用逆元素的性质 $x^{-1}x = e$ 得到

$$eg = eg'$$

由单位元的性质，有 $eg = g$ 和 $eg' = g'$，进而得到 $g = g'$，也就证明映射满足单射性。至于满射性，由 f 的定义有

$$f(h) = xh$$

因此，xH 中的任何元素均有 H 中的一个元素与之对应。这样我们就证明了 f 的双射性，也就是说 H 的所有左陪集与 H 本身都有相同个数的元素。

所以 G 就是 H 的所有不同左陪集（任何两个都不相交）的并集。设 i 是 H 的不同左陪集的个数，我们已经知道 H 的每个左陪集均有 $|H|$ 个元素，因此可以计算 G 中的元素个数，即有

$$|G|=\text{陪集基数的和}=i\times|H|$$

上面等式的两边都是整数，因此$|H|$整除$|G|$，定理得证。 ♣

习题

17.3.01 假设一个有限群G分别有一个阶为16的子群H和一个阶为35的子群K，证明$H\cap K=\{1\}$。

17.3.02 证明阶为101的群除了自身和{1}外没有其他子群。

17.3.03 证明阶为1009的群除了自身和{1}外没有其他子群。

17.4 子群的指标

我们在拉格朗日定理的证明中引入了陪集的概念，现在可以定义子群的指标。设G是一个群，H是G的一个子群，H在G中的**指标**就是G中H的（左）陪集的个数，记为$[G:H]$。

推论（拉格朗日定理的推论） 对一个有限群G及其子群H，$|G|=[G:H]\cdot|H|$。

证明 这个证明仅仅是拉格朗日定理证明过程中计数的简单重复，我们已证明G是H的左陪集的不相交的并集，而且每个陪集都有$|H|$个元素。因此，这个结论就是用另一种方法对G中的元素进行计数。 ♣

一个有关计数或整除性原则的结论是如下的子群指标的**乘法性质**。

命题 设G是一个有限群，H和I是G的子群，假设$H\supset I$，则有

$$[G:I]=[G:H]\cdot[H:I]$$

证明 群G是I的$[G:I]$个左陪集的不相交并集，同样它还是H的$[G:H]$个左陪集的不相交并集。如果我们能够证明H的任何左陪集是I的$[H:I]$个左陪集的不相交并集，则该命题的结论也就得以证明了。

设$gH=\{gh:h\in H\}$是H的一个左陪集，将H表示为I的$[H:I]$个左陪集的（不相交）并集，即

$$H=h_1I\cup h_2I\cup\cdots\cup h_{[H:I]}I$$

这样gH就可表示为

$$gH=g(h_1I\cup h_2I\cup\cdots\cup h_{[H:I]}I)=gh_1I\cup gh_2I\cup\cdots\cup gh_{[H:I]}I$$

上式右边当然是I的左陪集的并集。我们还需要验证$h_iI\cap h_jI=\varnothing(i\neq j)$即意味着

$$gh_iI\cap gh_jI=\varnothing$$

假设$g\in gh_iI\cap gh_jI$，则对某个$i_1\in I$和$i_2\in I$有$gh_ii_1=x=gh_ji_2$，两边同时左乘以g^{-1}则得到$h_ii_1=h_ji_2$。根据假设，h_ii_1是h_iI的元素，h_ji_2是h_jI的元素，但我们已经假设了$h_iI\cap h_jI=\varnothing$，所以这是不可能的。也就是说，我们证明了这个结论：如果$h_iI\cap h_jI=\varnothing$，则有$gh_iI\cap gh_jI=\varnothing$。这也就完成了我们对子群指标乘法性质的证明。 ♣

17.5 指数定律

应当指出的是，所谓的指数定律实际上并不是“定律”，而是指数符号的一些可证明的性质。指数符号本身也是非常基本的，无非就是重复乘法的一个缩写。

当然，我们必须确信明确指数符号g^n，这里n是整数，g是群G的一个元素。毕竟这仅仅是一个缩写：首先，

$$g^0 = e$$

并且

$$g^n = \underbrace{g \cdot g \cdot \cdots \cdot g}_{n} \quad (n \geqslant 0)$$

$$g^n = \underbrace{g^{-1} \cdot g^{-1} \cdot \cdots \cdot g^{-1}}_{|n|} \quad (n \leqslant 0)$$

还有一个更加准确但不太直观的定义 g^n 的方法，即**递归定义**：

$$g^n = \begin{cases} e & n = 0 \\ g \cdot g^{n-1} & n > 0 \\ g^{-1} \cdot g^{n+1} & n < 0 \end{cases}$$

这样定义便于计算和证明其他的结论。

下面我来验证一下所谓的指数定律：

命题 （指数定律） g 是群 G 的一个元素， m,n 为整数，则下面两个等式成立：

- $g^{m+n} = g^m \cdot g^n$
- $g^{mn} = (g^m)^n$

证明 最显然的是证明 $(g^{-1})^{-1} = g$ 。注意，我们绝对不能简单引用“指数定律”来证明这个命题。实际上，为了证明这个命题，我们必须认识到验证 x 的逆元素是 y 的方法就是去计算 xy 和 yx，看它们是否都等于单位元 e。因此为了证明 x^{-1} 的逆元素是 x，我们必须计算 $x^{-1}x$ 和 xx^{-1}，实际上根据逆元素的性质这两个都等于单位元 e。

这里还要用到 x 的逆元素的惟一性，让我们来证明这一点。假设 y 和 z 满足 x 的逆元素的性质，即 $yx = e$ 和 $xz = e$，给 $yx = e$ 两边右乘以 z 得到 $(yx)z = ez = z$，运用结合律以及条件 $xz = e$，将得到 $ye = z$，即 $y = z$ 。

我们可以用归纳法给出指数性质的证明，但这可能有一点冗长，也没有什么新意。我们就用穷尽的方法指明为什么这些“定律”是正确的。比如，我们来看一下，对非负整数 m,n，为什么 $g^{m+n} = g^m \cdot g^n$ 成立。

$$\begin{aligned} g^m \cdot g^n &= \underbrace{(g \cdot \cdots \cdot g)}_{m} \cdot \underbrace{(g \cdot \cdots \cdot g)}_{n} \\ &= \underbrace{(g \cdot \cdots \cdot g)}_{m+n} \\ &= g^{m+n} \end{aligned}$$

这里需要注意的是我们在推导过程中多次用到了结合律。因此这个指数“定律”最多也就是符号的性质。我们再看看为什么 $g^{mn} = (g^m)^n$ 成立：

$$\begin{aligned} (g^m)^n &= \underbrace{g^m \cdot \cdots \cdot g^m}_{n} \\ &= \underbrace{\underbrace{(g \cdot \cdots \cdot g)}_{m} \cdot \cdots \cdot \underbrace{(g \cdot \cdots \cdot g)}_{m}}_{n} \\ &= \underbrace{g \cdot \cdots \cdot g}_{mn} \\ &= g^{mn} \end{aligned}$$

正整数指数和负整数指数组合的情况可能要复杂一些，但方法是类似的。 ♣

17.6 循环子群

对群 G 的一个元素 g，令 $\langle g\rangle=\{g^n:n\in\mathbf{Z}\}$，称这个集合为**由 g 生成的 G 的循环子群**。

满足 $g^n=e$ 的最小正整数 n（如果存在）则称为 g 的**指数**或**阶**。群的元素 g 的阶通常记为 $|g|$，这里我们又用到了术语“阶”，但在后面我们会看到这些用法是一致的。

推论（指数定律的推论） 对群 G 的元素 g，G 的子集 $\langle g\rangle$ 的确是 G 的一个子群。

证明 结合律是很自然的。对群运算的封闭性和对求逆的封闭性可直接由指数定律得到，过程如下。首先，g^n 的逆元素刚好就是 g^{-n}，这是因为

$$g^n\cdot g^{-n}=g^{n+(-n)}=g^0=e$$

并且由 $g^m\cdot g^n=g^{m+n}$ 就直接验证了对乘法的封闭性。 ♣

定理 设 g 是群 G 的一个元素，n 是 g 的阶，则 g（作为群的元素）的阶等于 $\langle g\rangle$（作为子群）的阶。特别地，有 $\langle g\rangle=\{g^0,g^1,g^2,\cdots,g^{n-1}\}$。一般情况下，对于任意整数 i,j，

$$g^i=g^j \text{ 当且仅当 } i\equiv j \bmod n$$

证明 定理中的最后一个结论蕴涵着前面两个论断，所以我们只需证明最后一个论断。一方面，如果 $i\equiv j \bmod n$，则可以将 i 表示为 $i=j+\ell\cdot n$ 并计算（利用指数定律）：

$$g^i=g^{j+\ell n}=g^j\cdot(g^n)^\ell=g^j\cdot e^\ell=g^j\cdot e=g^j$$

另一方面，假设 $g^i=g^j$，不失一般性，如果必要时 i 和 j 可以互换，我们不妨假设 $i\leqslant j$，则由 $g^i=g^j$ 可得 $e=g^{j-i}$。利用约简/除法算法，可将 $j-i$ 表示为

$$j-i=q\cdot n+r$$

这里 $0\leqslant r<n$。由此即有 $e=g^{j-i}=g^{qn+r}=(g^n)^q\cdot g^r=e^q\cdot g^r=e\cdot g^r=g^r$。由于 n 是满足 $g^n=e$ 的最小正整数，因此必有 $r=0$，即 $n|(j-i)$，也就是说 $i\equiv j \bmod n$。 ♣

推论 g 是有限群 G 的一个元素，则 g 的阶 $|g|$ 整除群 G 的阶。

证明 我们已经证明了 $|g|=|\langle g\rangle|$，由拉格朗日定理，$|\langle g\rangle|$ 整除 $|G|$，结论得证。 ♣

习题

17.6.01 设 G 是一个群，$g,h\in G$，并且 $|g|=30,|h|=77$，证明 $\langle g\rangle\cap\langle h\rangle=\{e\}$。

17.6.02 证明群的元素与其逆元素有相同的阶。

17.6.03 不用计算证明在群 $\mathbf{Z}/100$（加法）中，元素1和99有相同的阶，11和89也有相同的阶。

17.6.04 假设对某个正整数 e 有 $x^e=1 \bmod n$，证明 $x^{-1} \bmod n$ 就等于 $x^{e-1} \bmod n$。

17.6.05 计算 2×2 实数矩阵群 $GL(2,\mathbf{R})$ 中两个行列式非零的元素 g,h 的阶：

$$g=\begin{pmatrix}0&-1\\1&0\end{pmatrix},h=\begin{pmatrix}0&1\\-1&-1\end{pmatrix}$$

并计算 g,h 的乘积 gh 以及 $(gh)^n$，n 为整数，然后证明 gh 是无限阶的。

17.6.06 设 G 是一个有限群，n 是 G 的元素的阶的最小公倍数，证明对所有的 $g\in G$，有 $g^n=e$。

17.6.07(*) 设 G 是一个阿贝尔群，m,n 是互素的正整数。令 g 是一个阶为 m 的元素，h 是一个阶为 n 的元素，证明 $|gh|=mn$。

17.6.08 x 是群 G 的一个元素，假定有 $x^{3\cdot 5}=e$ 且 $x^3\neq e$，证明 x 的阶为 5 或者 15。

17.6.09 证明满足 $1\leqslant i<11$ 的任意整数 i 都是整数模 11 的加法群 $\mathbf{Z}/11$ 的一个生成元。

17.6.10 验证 $\mathbf{Z}/8^{\times}$ 不能由单个元素生成。

17.6.11 证明，如果群 G 的元素 g 的阶为 n 并且 d 是 n 的一个因子，则 $g^{n/d}$ 的阶为 d。（类似地，g^d 的阶为 n/d）

17.7 欧拉定理

现在我们回到数论问题的讨论，作为拉格朗日定理的推论，我们将给出欧拉等式的一个简洁且概念化的证明，并且还要讨论指数定律和循环子群的问题。最后我们会给出欧拉定理相对精练的形式。

设 $\varphi(n)$ 是欧拉的 Φ 函数，它对满足 $0<\ell\leqslant n$ 且与 n 互素的整数 ℓ 进行计数。我们给出的证明是从欧拉原始的论据中提炼出来的。

定理 设 n 是一个正整数，$x\in\mathbf{Z}$ 且与 n 互素，则 $x^{\varphi(n)}\equiv 1 \bmod n$。

证明 整数模 n 且与 n 互素的集合 $\mathbf{Z}/n^{\times}$ 有 $\varphi(n)$ 个元素，由拉格朗日定理及其推论，这就意味着 $g\in\mathbf{Z}/n^{\times}$ 的阶 k 能整除 $\varphi(n)$，因此 $\varphi(n)/k$ 就是一个整数，于是

$$g^{\varphi(n)}=(g^k)^{\varphi(n)/k}=e^{\varphi(n)/k}=e$$

将这一结果运用到 $x-\bmod -n$ 上便得到定理的结果。 ♣

备注 这一方法同时也给出了费马定理的另一种证明，只不过此时 n 为素数，不用提及二项式系数。

进一步研究欧拉定理的证明，我们还有如下定理：

定理 设 n 是一个正整数，$x\in\mathbf{Z}$ 且与 n 互素，则满足 $x^{\ell}\equiv 1 \bmod n$ 的最小指数 ℓ 是 $\varphi(n)$ 的一个因子。也就是说 x 在乘法群 $\mathbf{Z}/n^{\times}$ 中的阶是 $\varphi(n)$ 的一个因子。

证明 与前面证明过程一样，x 的阶等于子群 $\langle x\rangle$ 的阶，由拉格朗日定理 x 的阶就是整个群 $\mathbf{Z}/n^{\times}$ 的一个因子。 ♣

17.8 群的指数

我们可以使欧拉定理的思想运用得更加准确和精练。对于一个群 G 以及对每个 $g\in G$，满足 $g^{\ell}=e$ 的最小正整数 ℓ 称为群 G 的**指数**。是否存在这样一个正整数 ℓ，单单从定义看还不明显。实际上，对于无限群 G，这样的 ℓ 不存在。但对于有限群，仅仅一个有限性就可以让我们来刻画指数。

命题 设 G 是一个有限群，则 G 的指数存在，而且 G 的指数=$|g|$ 的最小公倍数，$g\in G$。

证明 如果 $g^k=e$，则从我们前面对循环群的讨论可得，$|g|$ 能整除 k。另一方面，如果 $k=m\cdot|g|$，则 $g^k=g^{m|g|}=(g^{|g|})^m=e^m=e$。因为 G 是有限的，它的每一个元素的阶也是有限的。并且因为 G 中只有有限多个元素，那么它们的阶的最小公倍数 M 也是存在的。这样对所有的 $g\in G$，必有 $g^M=e$。因此，群 G 必有指数。如果对所有的 $g\in G$，都有 $g^k=e$，则 k 就能被 G 的所有元素的阶整除，当然就能被它们的最小公倍数整除。因此群 G 的指数实际上就等于其所有元素指数的最小公倍数。 ♣

这里我们还可以看到，拉格朗日定理还对群 G 的指数具备什么样的性质作出了限制。

推论 设G是一个有限群，则群G的指数整除G的阶$|G|$。

证明 由前面的命题，群G的指数等于G的所有元素的阶的最小公倍数。由拉格朗日定理，每一个这样的阶都是$|G|$的因子，一个固定数的所有因子的最小公倍数当然还是那个数的一个因子。 ♣

第18章 协议概述

在本章对协议的非正式和介绍性讨论中，特别重要的一点就是须记住攻击者的不同类型，至少我们要区别*被动的窃听者*和*主动的窃听者*之间的不同。在本书中几乎总是在讨论被动的窃听者，被动的窃听者虽然窃听却不会主动去中止通信，也不会去冒充发送方或接收方，或者同时冒充发送方和接收方。关于主动窃听者的一个更加重要的问题是他们的力量到底有多强大。**被动的窃听者**的力量还不足以去主动中止协议或者毁坏消息，但他们会窃听本不是发给他们的通信内容。通信中直接涉及的各方可能出于多种原因而实施欺骗行为，这就是**骗子**。骗子也根据其能力的不同分为**主动骗子**和**被动骗子**。在存在主动窃听者的情况下，要想防御能力非常强大的主动骗子就愈加困难。

许多协议只能保证（甚至是非正式地）成功抵御相关的被动窃听者。

在许多情况下，**预防**和**检测**之间是存在区别的。通常预防窃听或欺骗要比只检测这些攻击行为更加困难，因此任何实际的系统都必须包括当检测到攻击行为时如何防御的措施，而不是简单在佯装能够防止攻击。

18.1 基本的公钥协议

这也就是人们常说的公钥密码术的基本观点，但是（正如下面指出的那样）为了能够利用它，仍需要很多的知识。这里概述的结构将用于 RSA 公钥体制和 ElGamal 型公钥体制。

Alice 公开其公开密钥 K_{pub}，则任何想生成一个只能由 Alice 解读的明文 x 的人将用下式生成一个密文 y

$$y = E_{K_{\text{pub}}}(x)$$

然后这条消息通过不安全的信道传送给 Alice。假设除了 Alice 外没有任何人拥有解密密钥（私有密钥），并且假设密钥空间足够的大（并且没有弱点），那么只有 Alice 能够解密，当然解密时 Alice 使用了她的私有密钥。

这个结构中的一个明显的缺陷就是发送方可能伪装成另外一个人，而 Alice 没有任何简单的方法去检验这一点。因为她的公钥和模数是公开的，尽管只有 Alice 能读懂加密的消息，但任何人都可能发送消息。那么能够证明发送方身份的东西就是**签名**，这个签名包含着某些不可伪造的东西，它将证明没有任何其他人能够发送它。

另一个缺陷就是攻击者 Eve 可能首先**假冒** Alice，企图破坏协议结构，它会公布一个公钥，对应的私钥由她自己持有，而非 Alice。然后，如果 Eve 能够截取发给 Alice 的邮件，她就可以解密邮件而后以 Alice 的名义做出回复。同时 Eve 也不会让 Alice 看到原本发给她的邮件，Eve 甚至可以伪造某些似是而非的邮件，使 Alice 不会产生怀疑。在最近发表的一些文章中，人们发现 Eve 可以伪造发送方的公钥，然后自己截取并解密由 Alice 发送给发送方的加密消息。也就是说，Eve 使 Alice 和 Bob 之间的通信发生“短路”，对 Alice 而言，她伪装成了 Bob，对 Bob 而言，她则伪装成了 Alice。这实际上也是一种**中间人攻击**。

进一步深入讨论这个问题，我们将发现这实际上已变成了 Eve 和 Alice 相互竞争以“证明”谁是真正的 Alice。实际情况是消息的发送方可能只知道 Alice 是在线的一个人，Eve 假冒 Alice 之名所公布的公钥与 Alice 自己真正的公钥却是难以区别的。

认证公钥的一种方法就是利用**认证机构**（CA），这是一个可信赖的实体，它不会撒谎，它将检验某个公钥真正所属的特定实体。当然这还是一个不准确的概念。目前许多 WEB 浏览器已经使用了这样的认证机构，使所谓的安全交易依赖于认证证书。

另一种主动窃听涉及**时间安排**问题，一个主动的攻击者可以截取一条消息并在稍过一会后重新发送。或者类似地，一条消息稍后可以被复制和**重放**一次或多次。这些问题促使人们在消息中使用**时间戳**，这样收方就能够检测到这些攻击。但是网络的普遍不可预测性使这些问题比我们所希望的要复杂的多。

因此，仅有一个基本的公钥协议还是远远不够的。证书认证机构的解决方案仅是一种补救，但可能没有比这更好的方法了。

18.2 Diffie-Hellman 密钥交换

Diffie-Hellman 密钥交换既是最初的公钥密码的思想，也是目前所使用的一个重要机制。其实用的依据在于对称密码通常比非对称密码速度要快（对硬件和软件而言），因此公钥密码仅用来确立一个私有密钥，即**会话密钥**，它仅用于一次会话。

这个协议利用了离散对数的特点，当然还有相关的计算离散对数的困难性。因此，它不仅能在 $\mathbf{Z}/p$ 上离散对数的情形下运行，而且能够推广到任意的有限域、椭园曲线或者其上离散对数有意义的任意群结构上。

首先，Alice 和 Bob 协商一个大素数 m 和一个模 m 的本原根 g，这两个值不必保密，并且可以被一组用户共享。Alice 选择一个大的随机整数 x，秘密地计算 $X = g^x \% m$，并将 X 通过可能不安全的信道发送给 Bob。同时，Bob 选择一个大的随机整数 y，秘密地计算 $Y = g^y \% m$，并将 Y 通过可能不安全的信道发送给 Alice。

然后，Alice 秘密地计算 $k = Y^x \% m$，同样 Bob 秘密地计算 $k' = X^y \% m$。实际上，$k = k' \bmod m$，这是因为 $k' = X^y = (g^x)^y = g^{xy} = (g^y)^x = k \bmod m$。但是网络中的其他任何人都不能得到 k，除非他能计算离散对数。然后密钥 $k(=k')$ 就可以直接或间接地用作对称密码的密钥。

出于安全性的考虑，素数模数 m 应当很大，并且 m 应该是一个强素数，这就要求 $m-1$ 也应当被一个相当大的素数整除。我们可以想象 m 为 **Sophie Germain 素数**是一种较好的选择，这意味着 $(m-1)/2$ 也是素数。但是，这类素数相对于整个素数来说分布得比较稀疏。它们与**孪生素数**（如果 p 是素数，则 $p+2$ 也是素数）一样少。目前我们还不知道是否存在无限多个 Sophie Germain 素数（也不知道是否存在无限多个孪生素数）。

实际上协议并不真正需要 g 是一个本原根，仅仅（出于安全性考虑）要求它能生成 $\mathbf{Z}/n^x$ 的一个子群，使攻击者也不能计算相关的“对数”。

在这些协议的许多版本中，第一个版本容易受到主动攻击者的**中间人攻击**：一个主动的窃听者可能截取 Alice 发给 Bob 的消息以及 Bob 发给 Alice 的消息，她用自己的消息替换这些消息，并分别单独与 Alice 和 Bob 完成一个 Diffie-Hellman 交换，而且还维持了一种假像——Alice 和 Bob 直接进行了通信。如果 Alice 和 Bob 事前真的没有联系，并因此没有别的

方法相互验证对方的身份，则很难防止各种假冒攻击。

这个协议及其拓展由公钥合伙人申请了专利，但该专利已于 1997 年 4 月 29 日过期。

18.3 秘密共享

本节我们将给出一种数学机制，它保证在协议中只有当足够多的一组成员（不必是所有成员）达成一致时才能共享一个秘密。实现这个协议的机制称为**门限方案**。

下面我们来陈述秘密共享问题，在秘密共享的实际实现中，可能要附加上涉及公钥算法的指定协议以及相关的设备，具体是：给定一个在 t 个人 $A_1, A_2, \cdots, A_t$ 中 $k-$**共享的秘密** x，并给 A_i 分配部分信息 a_i，这样就有

- A_i 知道 a_i（但不是 a_j，对 $j \neq i$）
- 由任何 $k-1$ 个部分信息 a_i 中不能恢复出秘密 x 的任何一部分
- 由任何 k 个部分信息 a_i 很容易计算出秘密 x

也就是说，在**共享秘密**的 t 个实体中，至少有 k 个联合起来才能恢复秘密。对于一个给定的数 t 和一个给定的秘密 x，实现上述目标的一个部分信息清单 $a_1, a_2, \cdots, a_t$ 称为一个 (k,t) **门限方案**。这种方案最简单的例子使用了孙子定理（即中国剩余定理），具体如下：

设 $m_1, m_2, \cdots, m_t$ 是大于 1 且两两互素的整数，设 $a_1, a_2, \cdots, a_t$ 是整数，令 $M = m_1 m_2 \cdots m_t$，$M_i = M / m_i$，$n_i = M_i^{-1} \bmod m_i$。（因为对 $j \neq i$，m_i 和 m_j 互素，由因式惟一分解定理以及 M_i 是与 m_i 互素的多个整数的乘积，m_i 和 M_i 互素。因此其乘法逆元素存在且容易可由欧几里得算法计算得到）此时，作为孙子定理的一个简洁形式，如下 t 个同余式

$$x = a_i \bmod m_i，\text{对所有的 } i = 1, \cdots, t$$

等价于下面这个同余式

$$x = \sum_{i=1}^{t} a_i M_i n_i \mod M$$

对固定的 k，$1 \leqslant k \leqslant t$，令 H_k 为 k 个不同的 m_i 的最小乘积，h_{k-1} 表示 $k-1$ 个不同的 m_i 的最大乘积。比如，如果 $m_1 < m_2 < \cdots < m_t$，则

$$H_k = m_1 m_2 \cdots m_{k-1} m_k$$

$$h_{k-1} = m_{t-k+2} m_{t-k+3} \cdots m_{t-1} m_t$$

我们必须假设 H_k 大于 h_{k-1}，特别地，我们设 $H_k \geqslant (N+1) h_{k-1}$，$N$ 为某个非常大的正整数。

定理 对于任何可表示为区域 $h_{k-1} < x < H_K$ 内的一个数值的**秘密** x，令 $a_i = x\%m_i$，则集合 $\{a_1, a_2, \cdots, a_t\}$ 是 x 的一个 (k,t) 门限方案。

备注 如果模数 m_i 较大且相互接近，则安全性最好。这一点可由证明过程看出来。

证明 假设 $a_1, a_2, \cdots, a_k$ 已知，令 $M' = m_1 m_2 \cdots m_k$，$M_i' = M' / m_i$，$1 \leqslant i \leqslant k$。再令 $n_i' = M_i'^{-1} \bmod m_i$，$1 \leqslant i \leqslant k$。令

$$x' = \sum_{i=1}^{k} a_i M_i' n_i \bmod M'$$

则有（如同在孙子定理中）$x' = x \bmod M'$。因为 $M' \geqslant H_k > x$，秘密 x 已经模 M' 简化了，故秘密 x 可以用 $x = x'\%M'$ 来计算（这里不必假设 $m_1 < m_2 < \cdots < m_t$）。

另一方面，假设只有 $a_1, a_2, \cdots, a_{k-1}$ 是已知的，令 $M' = m_1 \cdots m_{k-1}$，$M_i' = M' / m_i$，$1 \leqslant i \leqslant k-1$。再令 $n_i' = M_i'^{-1} \bmod m_i$，$1 \leqslant i \leqslant k-1$。令

$$x' = \sum_{i=1}^{k-1} a_i M_i' n_i \bmod M'$$

同前面一样，因为 $M' = m_1 \cdots m_{k-1}$，所以

$$x' = x \bmod m_1 m_2 \cdots m_{k-1}$$

我们已经知道 $m_1 m_2 \cdots m_{k-1} \leqslant h_{k-1}$，因为 $h_{k-1} < x < H_k$ 并且（由假设）

$$(H_k - h_{k-1}) / h_{k-1} > N$$

则对 $y \bmod M'$ 至少存在 N 种可能性，故 $x' = y \bmod M'$。因此只有 $k-1$ 个 a_i 当然不足以还原出秘密 x。 ♣

为了构建使用这种数学机制的环境，我们可以假设存在一个**可信的中间机构**，它持有秘密 x，知道模数 m_i，能够计算门限方案数值 $a_1, \cdots, a_n$，它还能够通过安全的方式将 a_i 发送给实体 A_i。余下工作的就是构造一个协议，它允许 t 个实体中的 k 个共享他们信息而没有任何人行骗，或者至少没有检测到欺骗行为。这本身就是一个非常不简单的问题。

18.4 不经意传输

这一节我们来介绍另外一些也是比较常见的协议，在这些协议中没有相互信任的仲裁人。一个简单的例子就是，Alice 有一个秘密，她希望以这种方式传送给 Bob，即当协议完成之后，Alice 自己不知道 Bob 是否真正收到了这个秘密，但是 Bob 知道。

一个目的更明确也更复杂的这类协议是：Alice 持有多个秘密，并且希望将其中之一以这种方式传送给 Bob，即事后只有 Bob 知道秘密的确传过来了。如果 Bob 不想通过直接承认他不知道一个或多个秘密而使自己陷入尴尬，那么这就比较有意义的。

对于简单的情况，我们将假设这个简单的秘密是大数 $n = pq$ 的分解，这里 n 是两个大素数 p,q 的乘积，而且 p,q 满足 $p = 3 \bmod 4$ 和 $q = 3 \bmod 4$。这个假设是非常普通的，因为任何其他秘密可以用模数为 n 的 RSA 体制加密，那么只有知道 n 分解才能解密并揭露秘密。在稍微复杂一些的情况之中，我们将用离散对数的难解性来代替大数分解的困难性。

我们还要用到这样的结论，即**已知 x, y 满足 $x^2 = y^2 \bmod n$，但 $x \neq \pm y \bmod n$，则 n 可以被分解**。证明如下：由 $x^2 = y^2 \bmod n$ 可得 $(x-y)(x+y) = 0 \bmod n$，但是因为 $x \neq \pm y \bmod n$，则 $x-y$ 和 $x+y$ 模 n 都不为 0。即 n 能整除 $(x-y)(x+y)$，但不能整除 $x-y$，也不能整除 $x+y$。另一方面，因为 p 能整除 $(x-y)(x+y)$，则它必定整除 $x-y$ 和 $x+y$ 中的一个（或同时整除两个）。同理 q 也是这样。又因为 n 不能整除 $x-y$ 和 $x+y$ 中的单个因子，则必有 p 整除这两个因子中的一个而 q 整除另外一个。因此，利用欧几里得算法计算 $gcd(n, x-y)$ 将得到 p 或 q：实际，不论 $gcd(n, x-y)$ 是什么，它不能被 n 整除，但可以被 p 或 q 整除。因为 p 和 q 是素数，且 $n = pq$，则由惟一分解定理，不会有其他情形出现。这就证明了如果已知 x, y 满足 $x^2 = y^2 \bmod n$，但 $x \neq \pm y \bmod n$，则 n 可以被分解。 ♣

第一个协议中，Alice 将秘密传送给 Bob 而不知道 Bob 是否真正得到了它。秘密则是大数 $n = pq$ 的分解，这里 n 是两个大素数 p,q 的乘积，而且 p,q 满足 $p = 3 \bmod 4$ 和 $q = 3 \bmod 4$。

- Bob 选择一个随机数 x，$0 < x < n$，计算 $z = x^2 \bmod n$ 并将此值传送给 Alice；
- Alice 容易利用如下公式计算 z 模 p 的主平方根 w_1 和 z 模 q 的主平方根 w_2：

$$w_1 = z^{(p+1)/4} \bmod p$$

$$w_2 = z^{(q+1)/4} \bmod q$$

这两个公式来源于费马小定理。± 号可以随机选择，Alice 令 y_1 为 $\pm w_1$ 中的一个，y_2 为 $\pm w_2$ 中的一个，利用这些平方根，由孙子定理（和欧几里得算法）将求得一个 z 模 $n=pq$ 的平方根 y，使得 $y=y_1 \bmod p$ 和 $y=y_2 \bmod q$。Alice 将 y 传给 Bob。

- 如果 Bob 的选择的 x 满足性质 $x=\pm y \bmod n$，则他不会容易地分解 n。但是如果有

$$x=y_1 \bmod p \text{ 且 } x=-y_2 \bmod q$$

或者

$$x=-y_1 \bmod p \text{ 且 } x=y_2 \bmod q$$

则 Bob 就能够计算 $gcd(n, x-y)$ 以求得素数 p,q 其中之一。

由单个变量的多项式的惟一分解我们知道，一个二次方程 $y^2=z \bmod p$ 对素数 p 至多有两个根。因而由孙子定理（中国剩余定理），对不同的素数 p,q，$y^2=z \bmod pq$ 至多有四个根，实际上是对模 p 有两个根，模 q 有两个根，总共四个根。在上述协议的第三步，两个根 $\pm x \bmod n$ 就是四个根中的两个，Alice 可以相同的概率传送四个根中的任何一个。因此，在 Alice 的信息提供之后，Bob 就有 50% 的可能分解 n。但仍有 50% 的可能无法分解 n。

备注 协议完成之后，Alice 只知道 Bob 知道或不知道秘密的概率，尽管事实上只有两种可能，Bob 或者知道或者不知道秘密。问题的关键在于只有 Bob 知道自己是否掌握了秘密，而 Alice 不知道 Bob 是否知道了秘密。

现在我们假设 Alice 有两个秘密，她希望把它们传送给 Bob，但只能使 Bob 得到其中的一个，但 Alice 不知道 Bob 到底得到了那一个秘密。这里我们利用模素数 p 的离散对数问题的难解性。我们注意到，已知模数 p、模 p 的一个本原根 g 以及 $g^x \bmod p$ 和 $g^y \bmod p$ 的值，要想计算 $g^{xy} \bmod p$ 的值却不是一件容易的事情：由 $g^x \bmod p$ 和 $g^y \bmod p$ 的值计算 $g^{xy} \bmod p$ 的值似乎需要知道 x 和 y 的离散对数。我们假设秘密是两个整数 s_0, s_1，还可以假设这两个整数转换为二进制后长度相同，因为必要时可以填充1补齐。

- 随机选择一个大素数 p 并公开，而后再随机选择 **Z** 模 p 的一个本原根 g 以及一个在范围 $1<c<p-1$ 内的随机数 c；
- Bob 选择一个随机位 i 和一个随机数 x，$1<x<p-1$，以确立他的 ElGamal 型的公钥，他计算

$$\begin{aligned} b_i &= g^x \bmod p \\ b_{1-i} &= c \cdot g^{-x} \bmod p \end{aligned}$$

这里我们注意到无论随机位 i 是0还是1，随机位 $1-i$ 仍是0或1。Bob 用有序对 (b_0, b_1) 作为他的公开加密密钥，保密 (i,x)。

- Alice 验证 $b_0 b_1 = c \bmod p$；
- Alice 在区间 $[2, p-2]$ 上随机选择 y_0, y_1，并计算

$$\begin{aligned} a_0 &= g^{y_0} \quad t_0 = b_0^{y_0} \quad m_0 = s_0 \oplus t_0 \\ a_1 &= g^{y_1} \quad t_1 = b_1^{y_1} \quad m_1 = s_1 \oplus t_1 \end{aligned}$$

然后将 a_0, a_1, m_0, m_1 传送给 Bob。

- 由于随机位 i 不是公开的，Bob 将通过计算（均模 p）下式以得到秘密 s_i：

$$a_i^x = (g^{y_i})^x = (g^x)^{y_i} = b_i^{y_i} = t_i$$

$$s_i = m_i \oplus t_i$$

备注 因为 Bob 无法获得 c 的以 g 为底模 p 的对数，他就不能确定 b_0 和 b_1 的离散对数。Bob 因为不能计算 c 的离散对数，他也就不能获得 s_0 和 s_1。

18.5 零知识证明

作为一简单的例子，我们首先考查一下这样一个协议，Peter 想让 Vera 相信他知道大整数 n 的分解，但又不会把大数的因子泄露给 Vera，这里 n 是两个大素数 p,q 的乘积。

假设 Peter 知道 $n=pq$ 的分解，两个大素数 p,q 模 4 同余 3。

- Vera 随机选择整数 x 并将 $x^4\%n$ 发给 Peter；
- Peter 计算 x^4 模 p 的主平方根 $y_1=(x^4)^{(p+1)/2}$ 以及 x^4 模 q 的主平方根 $y_2=(x^4)^{(q+1)/2}$，利用孙子定理（和欧几里得算法）计算 y，使 $y=y_1 \bmod p$ 且 $y=y_2 \bmod q$。Peter 将这个值发回给 Vera。

备注 以模 4 同余 3 的素数为模数的平方根公式求得的主平方根，它本身是一个平方。实际，对于模 4 同余 3 的素数 p 以及 $a=b^2 \bmod p$，两个平方根是模 n 同余 $\pm b$ 的，并且确切地说 $\pm b$ 之一本身就是一个平方，因为 -1 是非平方的且 $\mathbf{Z}/p^{\times}$ 是循环的。

备注 因为 Vera 已经能够计算 x^2，Peter 当然没有泄露新的信息给 Vera。

备注 Vera 应当信服 Peter，因为 Peter 除了利用已知的因子 p,q 之外，没有别的方法计算平方根。况且在任何能够计算出模 n 平方根的方法，将会给出分解 n 的一个概率算法（这里 n 具有 $n=pq$ 的特殊形式，p,q 为不同的素数），具体如下：如果我们有一个 oracle（指某种不可解释的机制），它能计算模 n 的平方根，我们重复做如下步骤：随机选择一个 x，计算 $x^2\%n$，并把结果传送给 oracle，它将返回一个 x^2 模 n 的平方根 y。与前面讨论其他协议概念时所作的备注一样，因为非零平方模素数的两个平方根确实存在，由孙子定理，任何平方模 $n=pq$ 就存在四个平方根，$\pm x$ 就是其中的两个。设其余两个为 $\pm x'$，假设 x 的选择确实是随机的，则 oracle 就有 1/2 的概率将 $\pm x'$ 当作 y 返回。如果这样，那么 n 就不能整除 $x\pm y$（因为 $y\neq \pm x \bmod n$），但是 n 却能整除 x^2-y^2（因为 $x^2=y^2 \bmod n$）。因此 p 就能整除 $x\pm y$ 中的一个，而 q 则能整除另外一个。因此，$gcd(x-y,n)$ 就会是 p 或 q。因为 oracle 可以反复调用，在每一次调用中，获得分解的概率将会是 1/2。那么在 ℓ 次调用后，我们不能获得分解的概率将会是 $(1/2)^{\ell}$，当 l 趋于无穷时，这个概率值将为 0，我们将此看作是一个我们将在合理的时间内得到分解的一个指示。

18.6 鉴别

本节不同于本章的其他各节。在这里我们暂不考虑保密性问题，先简要考查在敌对环境下通信的可靠性问题，从相关但是不同的目标去考虑。实际上我们可以想象这种环境，对通信的阻碍要比失去秘密性更加严重。就目前的情形，我们不能假装能够明确地回答这些问题，只能指出这些问题。随着实践中加密技术的运用，公钥和私钥技术已经都被用来解决（获得了不同程度的成功）这里提出的问题。

消息鉴别 就是指用来检验消息来自于声称的来源并且没有被修改过的任何过程。更进一步的话，应当可能检验消息是以正确的顺序接收的，它们几乎是在发送之后就接收到的，没有任何丢失，等等。总之，对任何在各种敌对网络中传送的消息，除了秘密性之外，我们应当考虑检验：

- 消息的来源
- 消息的内容

- 消息的顺序
- 消息的时间
- 消息的接收
- 消息的非否认

来源 你肯定不希望攻击者以你的名义给你的联系人发送消息，而且你也不想让第三方能够假冒你信任的联系人给你发来伪造的消息。这种危险的一个重要例子就是：在许多邮件系统中，很容易在局域网上用任何假冒人希望的回复邮件地址给其他人发送电子邮件。

内容 即使一条消息的内容不需要保密，但你也不希望攻击者能够修改消息的内容。

顺序 你不想让攻击者删除一部分消息，或者让攻击者重新排列消息的顺序。

时间 攻击者不应当能够**重发**一个或多个旧的消息，似乎它们还是新的，或者**延迟**消息使它们比实际发送的时间晚一些。

接收 你也不希望攻击者能够错误地报告一条消息丢失了，或者错误地报告收到了一个消息。

非否认 你也不希望一个发送者能够否认曾经发送过消息给你。这个概念引发了环境的讨论，这里你可能不完全信任你的联系人。

当然，在上述每种情形下有两种可能：攻击的**阻止**和攻击的**检测**。阻止应该是理想的状态，但在目前的情形下（缺少量子信道），似乎检测是比较实际的目标。同样，由于所用网络的反复无常，消息的传送可能因为攻击行为而变得无序或者被延迟。因此，检验顺序和时间通常要比检验来源和内容更复杂。

尽管没有绝对出色的方法可以使用，但仍有三种不同类型的机制用于消息鉴别：

- **消息加密**
- **消息鉴别码**
- **散列函数**

消息加密 在采取这种方法的消息鉴别中，将利用密码技术对整个消息加密并发送密文，只有授权接收者才能解密。那么授权的接收者解密密文，实际上密文本身就是一个鉴别器，因为我们假设没有任何非授权者能够生成这样且能被正确解密的消息。

消息鉴别码 这个方法就是利用消息和秘密密钥的一个公开函数，生成一个小块头的鉴别信息。这一小块数据通常称为鉴别码（MAC），它随同明文一起发送。这里的鉴别器主体就是这个鉴别码的值。接收者重新计算所收到消息的鉴别码，并验证它是否与随消息发来的鉴别码匹配。

散列函数 这是没有密钥的消息鉴别码。如同鉴别码，散列函数也是利用一个公开散列函数生成一个小块头的鉴别数据。当然，尽管没有密钥改变了形式，并且似乎不受欢迎，在有些情形下要想配置一个共享秘密密钥也不大可能（不方便）。

当用加密的消息作为鉴别器时，其根据就是合法的发送者是惟一能够产生一个密文的人，该密文能被解密（合法的接收方也持有这个密钥）为相关明文。这有一点问题，解密的输入可能是任何东西，那么，如果合法的消息可以是相当任意的0,1字串，则解密将产生一个假定的明文，其合法性则很难验证。另一方面，如果希望明文是英文，则在没有真正的加密密钥的情况下要想伪造一个密文应当是困难的。

处理二进制消息时，可以附加一种结构使得合法的解密者很容易认识它，而没有解密密

钥时则无法复制或伪造它。比如，发送者可以在加密前附加消息的一个**校验和**（也称为**帧校验序列**，FCS）。那么这就需要合法的发送者和接收者均知道这种协议，解密者对整个消息解密，从明文的尾部去掉校验和，然后用得到的明文再计算一个校验和，看它是否与收到的校验和匹配。我们可以想象，一个闯入者要想生成一个伪造的密文，这个密文能被解密为明文加校验和，这将是比较困难的，除非他有加密密钥。尽管如此，这仍不能防止*重放攻击*。注意到在任何情况下，校验和（或任何其他结构）必须在加密前附加到明文上，这一点很重要。

加密本身不能防止旧消息的重放，旧消息可能被攻击者窃听或者复制。为了防止旧消息被重放，消息生成的时间应当被包含进来作为消息的一部分，这也就是**时间戳**。时间戳应当在其他任何鉴别码计算之前附加到消息上，同样也应在加密之前。

对消息不进行加密处理而计算一个 MAC 并附加到消息上，从计算上讲可能比全部加密要简便一些，这也允许接收者根据情况决定是否要进行验证。在计算 MAC 时使用秘密密钥算法可防止非法的攻击者修改消息并生成一个匹配的 MAC 值。

尽管消息的*接收方验证*的概念似乎看起来很好，但它却产生了一些问题，因为如果这种协议一旦被实现，则用发送未经请求的邮件的方法，可以不经许可获得其他人的一些信息。“查找器”协议（finger protocol）已允许这类行为，即使不允许验证上一次的登录时间，但这也产生了隐私问题。因此“查找器”协议经常是无效的，所以现在还没有广泛使用的接收方验证协议。

最后，非否认也是有问题的。某人给你发送了一个加密的消息，但稍后即声称他的密钥在那之前已经泄露了，所以他不对消息的内容负责。对此问题似乎没有简单的解决方法。

18.7 电子货币和电子商务

我们可能已经注意到“信息高速公路”的概念已经被“电子商务”和“电子银行”所取代。这也不应当感到惊奇，商业利益促使人们去开发借助于网络计算机的通信方式。但困难的问题随之而来：如何保护存储信用卡号码的服务器？如何与对方建立加密的连接？在本节，我们将指出这方面的一些问题，这些问题目前仍没有满意的解决方案。

消息的*非否认*问题引发了因特网上签署合同或者授权行为的问题。这也暴露出了网上的*身份*问题：你是谁？你声称你是某某，如何证明？并且可能出于不同的目的而保留不同的身份？

由于对医疗记录、信用记录和网上行为的记录（通常体现在浏览器的 *Cookie* 中）的关注，*数据隐私*引起了人们的兴趣。这种兴趣引发了这样一个问题：*如何完成网上的一次交易而不暴露你太多的信息*？另一方面，从商家的观点看，为了建立足够的信任以证明出售了商品，某些信息必须暴露。理想的状况是，购买者应当提供足够而不多余的信息给卖家，目的是要卖家确信交易完成了。类似地，根据因特网上大量诈骗存在的事实，卖家也需要确立信任。

在讨论电子交易时，一个比较重要的附带的技术问题是*故障模式*：如果一次电子交易在未完成时被中止将如何处理？你的钱有没有从银行提出来？可能已经从你的账户中提出来了，但没有到你的手上？确保交易*顺利进行*的协议、软件和硬件是很重要的。

将来，货币会实现真正的电子化。相对于使用纸币而言，使用传统信用卡的一个缺点就是，信用卡上的数值仅仅当它是与你的身份相联系时才是存在的。但你的交易却是可追踪的。

而纸币本身就有价值，不需要和你的身份发生联系，所以也不会产生其他交易的信息。目前的电子银行就遇到了如同信用卡一样的问题，比如某个人知道不少有关你资金交易的情况，在什么地方做了什么交易，你有什么样的消费模式。

理想的情况是，电子货币应有的价值应该独立于持有人的身份，它应当是可分割的，可转移的，不可重复使用的（也就是你不能把一张钱花两次），它还应当是不依赖于某个中间机构的，而且交易能够离线处理。

我们离这些目标还比较遥远。

第19章 环、域、多项式

我们以下即将定义的环和域的概念是数的各种概念的抽象。要掌握这些概念的目的部分是出于效率的考虑，因为我们不希望做一些不必要的重复。但一个普通的抽象可能建议使用另一个领域的观点，有时这会产生重要的发现。

这些抽象是逐步发展而来的，最先开始于 1800 年前后拉格朗日、高斯、艾森斯坦及其他一些人在数论方面的工作，当然包括伽罗瓦和阿贝尔在方程理论方面的工作。最初这些概念根本就不抽象，并且广泛伸入到计算问题中。到了 1900 年，人们对很多独立的现象进行了证实和分类，使得抽象的概念变得有意义并趋于一致。

19.1 环、域

环的概念是“数”的概念的推广，与**群**的概念相比，它可能还要更加直接一些。一个**环** R 是一个具有两个运算 $+$ 和 $\cdot$ 的集合，并且具有一个特殊的元素 0（**加法单位元**）和满足如下我们对加法和乘法期望或要求的性质：

- 加法**结合律**：对所有的 $a,b,c\in R$，$a+(b+c)=(a+b)+c$
- 加法**交换律**：对所有的 $a,b\in R$，$a+b=b+a$
- 对每个 $a\in R$，存在一个**加法逆元素**，记为 $-a$，且满足 $a+(-a)=0$
- 元素 0 对所有的 $a\in R$ 满足 $0+a=a+0=a$
- 乘法**结合律**：对所有的 $a,b,c\in R$，$a(bc)=(ab)c$
- 乘法和加法满足左右**分配律**：即对所有的 $a,b,c\in R$，$a(b+c)=ab+ac$，$(b+c)a=ba+ca$

我们写这里的乘法时，就像在高中代数中的一样，通常略去点而直接写为 $ab=a\cdot b$。

通常一个特定的环有如下加法的特性或性质：

- 如果在一个环中存在一个元素 1，且对所有的 $a\in R$ 满足性质 $1\cdot a=a\cdot 1$，则称 1 为环的**乘法单位元**或**单位**，那么该环被称为**有一个单位元素**或**单位元**，或者称为**有单位元的环**。1 是环的单位元，我们还要求在一个环中 $1\neq 0$。
- 如果在一个环 R 中，对所有的 $a,b\in R$，有 $ab=ba$，则环 R 称为**交换环**。当且仅当乘法是交换的，一个环是交换环。

通常，但不总是这样，我们感兴趣的环是有单位元 1 的环。乘法交换的条件通常是满足的，但也有例外，如矩阵乘法就是非交换的。

- 在一个有单位元 1 的环中，对于给定的元素 $a\in R$，如果存在 $a^{-1}\in R$ 使得 $a\cdot a^{-1}=1$ 且 $a^{-1}\cdot a=1$，则 a^{-1} 称为是 a 的**乘法逆元素**。如果 $a\in R$ 有乘法逆元，则 a 称为环 R 的**一个单位**，环 R 中所有单位的集合记为 $R^{\times}$，称为环 R 的**单位群**。
- 每个非零元素都是单位的交换环称为**域**。
- 每个非零元素都是单位的非交换环称为**商环**。

- 环 R 中的一个元素 r，若对某个非零的 $s\in R$ 满足 $r\cdot s=0$ 或者 $s\cdot r=0$，则称 r 为一个**零因子**。一个没有非零零因子的交换环称为**整环**。
- 交换环 R 具有**消元性质**，即对任意的 $r\in R, r\neq 0$，如果对 $x,y\in R$ 且 $rx=ry$，则有 $x=y$。我们所熟悉的许多环具备这样的性质。

备注 在使用单位一词时实际上出现了不一致的情况。但这只是一个用词方式的问题。所以单位元是1，单位 a 只是一个有乘法逆元素的元素。当然，除非有一个单位（意味着存在1），否则就没有逆元素。我们总是能够从上下文中知道它具体指的是什么。

我们应当能够认识到记号 $-a$（加法逆元素）和 a^{-1}（乘法逆元素）跟通常意义上的“负 a”和“被 a 除”是一致的，这一点很重要。但目前我们还不能明确地认为这就是高中代数的东西。我们必须证明所有这些平常的东西在抽象的结构中仍然是正确的。

如果我们只考虑环 R 中的加法和0，则我们就得到了一个阿贝尔群，称为 R 的**加法群**。

环中单位的集合 $R^{\times}$ 加上单位元当然是一个群，其单位元就是1。如果环 R 的乘法是交换的，则这个群就是阿贝尔群。

在下面我们给出的例子中，还有一些更加实用的术语，通常一个群就是指环的加法群或者单位群。有许多例子不具备这些性质，但也有一些基本的例子具备这些性质。

整数集合 $\mathbf{Z}$ 在通常的加法和乘法运算下构成一个环。这个环当然是交换的，其乘法单位元为1，单位群 $\mathbf{Z}^{\times}$ 刚好是 $\{\pm 1\}$。这个环是整环。

整数中偶数的集合 $2\mathbf{Z}$ 在通常的加法和乘法运算下，构成一个没有单位元的交换环。正如这个例子所揭示的那样，通常在一个环中没有单位元在某种程度上是可以人为的，因为有一个“较大”的环，它有单位元，在这个环中却没有单位。

整数模 m 的集合 $\mathbf{Z}/m$ 构成一个有单位元的交换环。单位群正如记号所表示的那样，为 $\mathbf{Z}/m^{\times}$：注意这个单位群的记号我们前面已经用过了。

设 p 为一个素数，则整数模 p 的集合 $\mathbf{Z}/p$ 构成一个域，因为所有的小于 p 的正整数都有一个模 p 的乘法逆元素（由欧几里得算法即可求得）。单位群记为 $\mathbf{Z}/p^{\times}$。

所有 $n\times n$（n 固定）实数矩阵的集合，在通常的矩阵乘法和加法下构成一个环。除了 $n=1$ 的平凡情况以外，这个环是非交换的。单位群为群 $GL(n,\mathbf{R})$。

有理数集 $\mathbf{Q}$、实数集 $\mathbf{R}$ 和复数集 $\mathbf{C}$ 都是域的例子，因为它们所有的非零元素都有乘法逆元素。

正如我们一开始讨论群的性质一样，我们也应当考查一下环具备哪些性质。根据我们以前对数的一些经验，通常做出一些假定也是合理的。但是最好对这些性质给出一些简短的证明，而不是在意识上理所当然地就接受它们。

设 R 是一个环，我们将证明如下一些基本的性质：

- 加法单位元的惟一性：如果存在一个元素 $z\in R$ 和另一个元素 $r\in R$，使得 $r+z=r$，则 $z=0$。（注意，我们这里需要的条件只是一个元素 $r\in R$，而不是对所有的 $r\in R$）
- 加法逆元的惟一性：固定 $r\in R$，如果存在 $r'\in R$，使得 $r+r'=0$，则实际上有 $r'=-r$，即 r 的加法逆元。
- 乘法单位元的惟一性：假设 R 有单位元1，如果存在 $u\in R$，使得对所有的 $r\in R$ 满足 $u\cdot r=r$，则 $u=1$。或者如果对所有的 $r\in R$ 满足 $r\cdot u=r$，则 $u=1$。实际上，我们需要的就是 $1\cdot u=1$ 或 $u\cdot 1=1$ 来确保 $u=1$。

- 乘法逆元的惟一性：如果$r\in R$有一个乘法逆元r^{-1}，并且如果$r'\in R$使得$r\cdot r'=1$，则$r'=r^{-1}$。或者假设$r'\cdot r=1$，仍然能够得到$r'=r^{-1}$。
- 对于$r\in R$，我们有$-(-r)=r$。即r的加法逆元的加法逆元仍是r。

证明（加法单位元的惟一性） 如果存在一个元素$z\in R$和$r\in R$，使得$r+z=r$，给这个等式左右两边同时加上$-r$，由加法逆元的定义则有

$$(r+z)-r=r-r=0$$

再利用加法的交换律和结合律以及0的性质，上式左边为

$$(r+z)-r=(z+r)-r=z+(r-r)=z+0=z$$

左右两边对照起来就有$z=0$，这就证明了我们的结果。 ♣

证明（加法逆元的惟一性） 固定一$r\in R$，如果存在$r'\in R$，使得$r+r'=0$，给这个等式左右两边同时加上$-r$则有

$$(r+r')-r=0+(-r)$$

利用加法的交换律和结合律，上式左边就变为

$$(r+r')-r=(r'+r)-r=r'+(r-r)=r'+0=r'$$

而右边为$0+(-r)=-r$，因此$r'=-r$，结论得证。 ♣

证明（乘法单位元的惟一性） 假设$1\cdot u=1$或者$u\cdot 1=1$就能确保$u=1$。由已知条件和乘法单位元的性质很容易得到我们的结论。 ♣

证明（乘法逆元素的惟一性） 假设$r\in R$有一个乘法逆元r^{-1}，并且如果$r'\in R$使得$r\cdot r'=1$，我们同时给等式两上边左乘以r^{-1}，再利用1的性质则有

$$r^{-1}\cdot(r\cdot r')=r^{-1}\cdot 1=r^{-1}$$

利用乘法的结合律，以及乘法逆元和单位元的性质，上式左边为

$$r^{-1}\cdot(r\cdot r')=(r^{-1}\cdot r)\cdot r'=1\cdot r'=r'$$

两边结合就有$r'=r^{-1}$。 ♣

最后一个结论$-(-r)=r$的证明与群的类似结论的证明一样，逆元素的逆元素就是原来的元素。

我们在高中甚至更早就知道了这样的结论：如“减减得加”、“零乘任何数得零”。从环的一些公理来看这些结论是很有趣的，我们也可以证明这些结论。

这些结论比前面的一些“显然”的结论稍有不同，它们涉及乘法和加法的交叉运算。这些简单的证明是我们证明一些关于环的一般结论的很好模板。

设R是一个环，则

- 对任意的$r\in R$，$0\cdot r=r\cdot 0=0$；
- 假设R中存在1，设-1是1的加法逆元，则对任意的$r\in R$，有$(-1)\cdot r=r\cdot(-1)=-r$，这里$-r$表示$r$的加法逆元；
- 设$x,y\in R$，且$-x,-y$分别是x,y的加法逆元，则$(-x)\cdot(-y)=x\cdot y$。

证明 在下面的讨论中，我们需要记住，要想证明$b=-a$，就只需证明$a+b=0$。

现在我们来证明“零乘任何元素得零”这一结论。设$r\in R$，则

$$\begin{aligned}0\cdot r&=(0+0)\cdot r \quad (\text{因为}0+0=0)\\&=0\cdot r+0\cdot r \quad (\text{分配律})\end{aligned}$$

给两边同时加上$-(0\cdot r)$则有

$$0=0\cdot r-0\cdot r=0\cdot r+0\cdot r-0\cdot r=0\cdot r+0=0\cdot r$$

这就证明了 $0\cdot r=0$，$r\cdot 0=0$ 的证明与此类似。

我们再来证明 $(-1)\cdot r=-r$，我们所要证明的就是 $(-1)r$ 是 r 的加法逆，而且知道它是惟一的。所以我们只需要验证 $r+(-1)r=0$ 即可

$$r+(-1)r=1\cdot r+(-1)\cdot r=(1-1)\cdot r=0\cdot r=0$$

这里利用了1的性质以及分配律和我们已经证明的结果 $0\cdot r=0$。

最后我们来证明 $(-x)\cdot(-y)=x\cdot y$，因为我们已经知道 $-(-r)=r$，所以我们要证明的结论就可变为 $(-x)\cdot(-y)=-(-(x\cdot y))$。我们分两步来证明它：首先我们需证明 $-(xy)=(-x)y$，这可由下面的计算得到：

$$(-x)y+xy=(-x+x)y=0\cdot y=0$$

将这两个方面结合起来，那么我们只需要再证明 $(-x)(-y)+(-x)y=0$，而这可由下面的计算得出：

$$(-x)(-y)+(-x)y=(-x)(-y+y)=(-x)\cdot 0=0$$

这里用到了分配律和性质 $r\cdot 0=0$。这就完成了我们的证明。 ♣

习题

19.1.01 验证模 m 为1的同余类 $\overline{1}$ 是环 $\mathbf{Z}/m$ 的乘法单位元“1”。

19.1.02 验证 $\mathbf{Z}/6$ 的子集 $\{\overline{0},\overline{3}\}$ 是一个环，并且 $\overline{3}$ 是乘法单位元“1”。

19.1.03 验证 $\mathbf{Z}/10$ 的子集 $\{\overline{0},\overline{2},\overline{4},\overline{6},\overline{8}\}$ 是一个环，并且 $\overline{6}$ 是其乘法单位元“1”。

19.1.04 验证 $\mathbf{Z}/15$ 的子集 $\{\overline{0},\overline{3},\overline{6},\overline{9},\overline{12}\}$ 是一个环，并且 $\overline{6}$ 是其乘法单位元“1”。

19.1.05 求环 $\mathbf{Z}/4$, $\mathbf{Z}/5$, $\mathbf{Z}/6$ 中的单位群。

19.1.06 求环 $\mathbf{Z}/12$ 中的单位群。

19.1.07 验证所有偶整数集合 $2\mathbf{Z}$ 是一个环，尽管它没有单位元。

19.1.08 验证如果 n 是合数，则 $\mathbf{Z}/n$ 有非零零因子。

19.1.09 验证如果 p 是素数，则 $\mathbf{Z}/p$ 是整环。

19.1.10 证明环中的一个乘法可逆元素不是零因子。（注意 $1\neq 0$）

19.1.11 设 R 是形如 $a+bi$ 的数的集合，其中 $a,b\in\mathbf{Q}$ 且 $i=\sqrt{-1}$。做为练习，验证 R 对加法和乘法是封闭的，若 R 是一个环，进而证明它是一个域。（提示：分母有理化）

19.1.12 设 R 是形如 $a+b\sqrt{2}$ 的数的集合，其中 $a,b\in\mathbf{Q}$。做为练习，验证 R 对加法和乘法是封闭的，若 R 是一个环，进而证明它是一个域。（提示：分母有理化）

19.1.13 设 R 是形如 $a+bi$ 的数的集合，其中 $a,b\in\mathbf{Z}$ 且 $i=\sqrt{-1}$。验证 R 对加法和乘法是封闭的，若 R 是一个环，找出其单位群。

19.1.14 设 R 是形如 $a+b\sqrt{-5}$ 的数的集合，其中 $a,b\in\mathbf{Z}$。验证 R 对加法和乘法是封闭的，若 R 是一个环，找出其单位群。

19.1.15 证明在环中，只有当 $r=0$ 时，才有等式 $r+r=r$ 成立。

19.1.16 在环 $\mathbf{Z}/15$ 中求几个元素 x,y 非零而它们的积为零的例子。

19.1.17 在环 $\mathbf{Z}/21$ 中求几个元素 x,y 非零而它们的积为零的例子。

19.1.18 在环 $\mathbf{Z}/16$ 中求几个元素 x,y 非零而它们的积为零的例子。

19.2 整除性

在试图证明各种交换环“具有惟一分解”之前，我们应当明确其含义是什么。为了明确这一点，我们需要再次用到整除性。我们假设在本节所讨论的环是交换的且有一个单位元。

第一个需要掌握的就是消去律可能不成立，而这一点则与非零零因子出现密切相关。

定理 一个交换环 R 具有消元的性质当且仅当它是一个整环。

证明 假设 R 具有消元的性质，并假设 $r\cdot s=0$，因为 $r\cdot 0=0$ 对任何 $r\in R$ 成立，所以我们有 $r\cdot s=r\cdot 0$。如果 $r\neq 0$，则消去 r 得到 $s=0$。同样，如果 $s\neq 0$，则得到 $r=0$。这就表明消去律成立意味着该环没有非零零因子，即 R 是整环。

另一方面，假设 R 是整环，即它没有非零的零因子，并且假设 $ra=rb$，且 $r\neq 0$，则我们有 $r(a-b)=0$，因为 $r\neq 0$，则必有 $a-b=0$，即 $a=b$，这就得到了我们所希望的消去律。 ♣

在一个交换环 R 中，称 $x\in R$ **整除** $y\in R$，如果存在 $z\in R$，使得 $y=zx$。我们还称 y 是 x **的倍数**。如同普通的整数，我们用 $x\mid y$ 表示 x 整除 y。x 就可以称为 y 的一个**因子**。如果 $xz=y$ 并且 x 和 z 都不是单位，则称 x 为 y 的一个**真因子**。

我们还要记住，任何元素整除 0，这是因为 $r\cdot 0=0$。但是 0 只能整除自己。另一方面，如果 $u\in R$ 是 R 的一个单位（即它有一个乘法逆元），则它整除每个元素：设 $r\in R$ 为任意元素，我们可由 $r=r\cdot 1=r\cdot(u^{-1}\cdot u)=(r\cdot u^{-1})\cdot u$ 得知 r 是 u 的倍数。

R 的一个元素 p 称为是**素元或不可约元**，如果 p 本身不是 R 中的单位，而且如果 $xy=p$（$x,y\in R$），则 x 或者 y 是 R 中的一个单位。这个定义的另一种说法会是：一个元素是素元当且仅当它没有真因子。

- 如果 d 是整环 R 中一个非零元素 r 的真因子，则有 $R\cdot r\subset R\cdot d$，但 $R\cdot r\neq R\cdot d$。

证明 因为 d 是 r 的一个因子，则存在 $x\in R$，使得 $xd=r$。由 R 对乘法的封闭性，则有

$$R\cdot r=R(xd)=(Rx)d\subset R\cdot d$$

假设 $R\cdot r=R\cdot d$，则 $d=1\cdot d\in R\cdot d=R\cdot r$，因此必存在一个 $s\in R$，使 $d=s\cdot r$。但是

$$r=xd=x(sr)=(xs)r$$

即 $r(1-xs)=0$。因为 $r\neq 0$ 且 R 是整环，则必有 $1-xs=0$，即 $xs=1$。这也就是说 x 是一个单位，这就与假设条件 $xd=r$ 是 r 的一个真分解相矛盾。 ♣

两个素元 $p,q\in R$ 称为是**相伴的**，如果存在一个单位 $u\in R$，使得 $q=up$。（因为单位的逆仍是单位，此条件是对称的，即它等价于存在一个单位 $v\in R$，使得 $p=vq$。）

出于将元素分解为素元的目的，两个相伴的素元通常我们将其视为同一个元素。

习题

19.2.01 证明在环 $\mathbf{Z}/35$ 中所谓的消去律不成立：即对这样的模数，找出（非零）元素 $a,b,c\in\mathbf{Z}/35$，使得 $ca=cb$，但 $a\neq b$。

19.2.02 证明在环 $\mathbf{Z}/77$ 中所谓的消去律不成立：即对这样的模数，找出（非零）元素 $a,b,c\in\mathbf{Z}/77$，使得 $ca=cb$，但 $a\neq b$。

19.2.03 n 是一个合数，证明在环 $\mathbf{Z}/n$ 中所谓的消去律不成立：即对这样的模数，找出（非零）元素 $a,b,c\in\mathbf{Z}/n$，使得 $ca=cb$，但 $a\neq b$。

19.2.04 设 u 是交换环 R 中的一个单位，证明在 R 中不存在非单位的元素整除 u。

19.2.05 证明在环中如果 $x=yu$ 且 u 是一个单位，则对某个单位 u' 有 $y=xu'$。

19.2.06 设 R 是一个有单位1的整环，并且对 $x,y\in R$ 有 $Rx=Ry$，证明存在一个单位 $u\in R^{\times}$，使得 $y=ux$。

19.2.07 假设在一个整环 R 中有 $xy=z$，并且 x 和 y 都不是单位，证明 $Rz\subset Rx$，但是 $Rz\neq Rx$。

19.3 多项式环

另外一类重要的且是常见的环结构就是**多项式环**：设 R 是一个有单位的交换环，定义

$$R[x]=\{\text{系数为}R\text{中的多项式}\}$$

其上的运算则为多项式的加法和乘法。我们特别感兴趣的是那些系数在某个域 k 中的多项式。通常我们默认的这个域是 **Q**、**R** 或者 **C**，有时我们也允许这个域是 $\mathbf{Z}/p$（p 为素数）这样的有限域。

设 R 是一个有单位1的交换环，并设 x 是我们通常所说的“变量”或“不定元”，则系数属于 R 的 x 的**多项式环**是这样一种结构：使用变元 x 且系数属于环 R 的所有多项式集合。我们也可称它为 R 上 x 的多项式环，使用方括号 $R[x]$ 表示 k 上 x 的多项式环。在多项式的通常加法和乘法运算下，$R[x]$ 是一个环。

我们已经非常熟悉在实数域或复数域上的多项式，这里也没有什么特别的地方。一个**变元**为 x 的多项式可表示为如下形式：

$$P(x)=c_nx^n+c_{n-1}x^{n-1}+\cdots+c_3x^3+c_2x^2+c_1x+c_0$$

其**系数**就是环 R 中的“数” $c_n,\cdots,c_0$，**常数项**为 c_0。如果 $c_n\neq 0$，则称 c_nx^n 为**最高次项**或**首项**，c_n 就称为**最高次项的系数**或**首项系数**。

直和项 c_ix^i 为 i **次项**，有时 i 也被称为直和项 c_ix^i 的**阶**，最高次项的阶称为多项式的**次数**。

备注 我们用上述形式表示多项式时总是忘记声明 c_n 是否非零。有的时候我们以上述方式表示多项式，已假定 c_n 非零。

如果一个多项式的首项或最高次项的系数为1，则称其为首一多项式。

备注 有时我们总是想当然地认为多项式是一种函数，实际上这是有问题的。多项式可以产生函数，但它们本身仅仅是多项式而已。准确的表示应当是：系数属于环 R 的一个多项式

$$f(x)=c_nx^n+c_{n-1}x^{n-1}+\cdots+c_1x+c_0$$

生成一个环 R 上的函数，记为

$$f(a)=c_na^n+c_{n-1}a^{n-1}+\cdots+c_1a+c_0,\quad a\in R$$

也就是用 a 代替变元 x。这是一个 R 到 R 的函数。

如果我们错误地把多项式当作函数，则多项式本身所具有的一些特性就消失了。比如，我们来考查多项式环 $(\mathbf{Z}/2)[x]$ 中的多项式 $f(x)=x^3+x^2+x+\overline{1}$，其系数属于 $\mathbf{Z}/2$，则

$$f(\overline{0})=\overline{0}^3+\overline{0}^2+\overline{1}+\overline{1}=\overline{0}$$

$$f(\overline{1})=\overline{1}^3+\overline{1}^2+\overline{1}+\overline{1}=\overline{0}$$

这个由多项式生成的函数是一个0-函数，但显然多项式却不是零多项式。还有一个例子，考虑 $(\mathbf{Z}/3)[x]$ 中的多项式 $f(x)=x^3-x$，计算得 $f(\overline{0}),f(\overline{1}),f(\overline{2})$ 均为 $\overline{0}$，但多项式本身显然为非零多项式。

习题

19.3.01 求 $x^2+3x+2=0$ 在 $\mathbf{Z}/5$ 中的两个根。

19.3.02 求 $x^2+3x+2=0$ 在 $\mathbf{Z}/7$ 中的两个根。

19.3.03 求 $x^2+1=0$ 在 $\mathbf{Z}/5$ 中的两个根。

19.3.04 求 $x^2+1=0$ 在 $\mathbf{Z}/13$ 中的两个根。

19.3.05 验证 $x^2+x+1=0$ 在 $\mathbf{Z}/2$ 中没有根。

19.3.06 设 p 是一个素数，验证 $x^p-x+a=0$ 在 $\mathbf{Z}/p$ 中没有根。

19.3.07 求 $x^2-1=0$ 在 $\mathbf{Z}/8$ 中的四个根。

19.3.08 求 $x^2-2=0$ 在 $\mathbf{Z}/161$ 中的四个根。

19.3.09 求 $x^2-1=0$ 在 $\mathbf{Z}/105$ 中的八个根。

19.3.10 求 x^2-1 在 $\mathbf{Z}/15$ 中除显然的 $(x-1)(x+1)$ 之外的另一种分解。

19.3.11 求 x^2-2 在 $\mathbf{Z}/161$ 中的两个不同的因子为首一的分解式。

19.3.12 设 $k[x]$ 是域 k 上变元为 x 的多项式环，求该环的单位群 $k[x]^\times$。

19.3.13 设 $R[x]$ 是有单位1且交换的环 R 上变元为 x 的多项式环，求该环的单位群 $R[x]^\times$。

19.3.14(*) 为什么二次公式不能求出 $x^2+x+1=0$ 在 $\mathbf{Z}/2$ 中的根？

19.4 欧几里得算法

在域 k 上的多项式环 $k[x]$ 也存在一个**除法算法**，并因此存在一个与通常的整数环 $\mathbf{Z}$ 上形式相同的**欧几里得算法**。除法算法就是用一个多项式去除另一个多项式，产生一个余项，跟我们在高中甚至更早学到的一样。只不过那时我们掌握的除法过程不依赖系数域，当然余项的次数要低于除式的次数。

例如，我们用 x^2+1 来除 x^3+1：$(x^3+1)-x\cdot(x^2+1)=x-1$，这已是最后的约简形式，因为 $x-1$ 的次数已严格低于 x^2+1 的次数。我们再来用 x^2+1 约简 x^5+1，步骤如下：

$$(x^5+1)-x^3\cdot(x^2+1)=-x^3+1$$

$$(-x^3+1)+x\cdot(x^2+1)=x+1$$

两个步骤合起来则有 $(x^5+1)-(x^3-x)\cdot(x^2+1)=x+1$。

因为除法算法对域上的多项式也成立，作为一个推论我们也有一个欧几里得算法：至关重要的一点是，在数的欧几里得算法中，除法算法的每一步都得到一个更小的数。

我们来做一个 $\mathbf{Z}/2$ 上多项式的除法，用 x^3+x+1 除 $x^7+x^6+x^5+x^4+x^3+x^2+1$，详细步骤如下：

$$\begin{array}{r|cccccccc}
 & x^4 & +x^3 & +0 & +x^1 & +0 & Rx^1 & +x^0 & \\
x^3+0+x^1+x^0 & x^7 & +x^6 & +x^5 & +x^4 & +x^3 & +x^2 & +0 & +x^0 \\
 & x^7 & +0 & +x^5 & +x^4 & +0 & +0 & +0 & +0 \\
\hline
 & & x^6 & +0 & +0 & +x^3 & +x^2 & +0 & +x^0 \\
 & & x^6 & +0 & +x^4 & +x^3 & +0 & +0 & +0 \\
\hline
 & & & & x^4 & +0 & +x^2 & +0 & +x^0 \\
 & & & & x^4 & +0 & +x^2 & +x^1 & +0 \\
\hline
 & & & & & & & x^1 & +x^0
\end{array}$$

用一个式子表示则是 $(x^7+x^6+x^5+x^4+x^3+x^2+1)-(x^4+x^3+x)(x^3+x+1)$

$$=x+1$$

如果系数域不是 **Z**/2，则容易出现一个因子 D 不是**首一**的情形，也就是其首项系数 c_n 不是 1。此时我们多项式 $c_n^{-1}D$ 是首一的，则可用 $c_n^{-1}D$ 去除。由除式为 $c_n^{-1}D$ 的除法算法表达式 $F=Q\cdot(c_n^{-1}D)+R$，我们可以直接得到除式为 D 的表达式：

$$F=(Qc_n^{-1})\cdot D+R$$

非首一多项式的除法中数的运算与前面的方法一样，只不过采用这种方法可以避免错误（特别是在手工计算时）。

分解、不可约性检验：就像在通常的整数分解方法中一样，对于较小的有限域上的小次数多项式，我们也可使用一个直接的分解算法。类似于普通整数的分解 $|xy|=|x|\cdot|y|$，对一个域上多项式的分解我们也需要这样一个事实：

$$\deg(P\cdot Q)=\deg P+\deg Q$$

这里的 deg 是多项式的次数。这个结论的成立依赖于这个事实：域中没有零因子。令：

$$P(x)=a_mx^m+a_{m-1}x^{m-1}+\cdots+a_2x^2+a_1x+a_0$$

$$Q(x)=b_nx^n+b_{n-1}x^{n-1}+\cdots+b_2x^2+b_1x+b_0$$

显然这里的首项系数 a_m,b_n 均不为零。在乘积 $P\cdot Q$ 中最高次项是 x^{m+n}，它只可能由 P 和 Q 的首项相乘而来，因此它的系数为 $a_m\cdot b_n$。又因为 P 和 Q 的首项系数都不为零，而且域中非零元素的乘积仍非零，所以 x^{m+n} 的系数非零。这就证明了**乘积的次数为次数的和。**

- 对于多项式 F 的一个真因子 D，$0<\deg D<\deg F$。更进一步，如果 F 有一个真因子，则它有一个真因子 D 满足 $0<\deg D\leqslant\frac{1}{2}\deg F$。

证明　如果 F 的一个真因子 D 的次数大于等于 $\frac{1}{2}\deg F$，则 F/D 的次数小于等于 $\frac{1}{2}\deg F$。 ♣

有关多项式的分解还有如下一些简单的结论：

- 每个线性多项式是不可约的，因为不存在次数介于1和0之间的多项式；
- 如果一个二次多项式可以真分解，则它必定是两个线性因子的乘积；
- 如果一个三次多项式可以真分解，则它必定至少有一个线性因子；
- 如果一个四次或更高次多项式可以真分解，则它可能没有线性因子；
- 系数域为 k 的多项式 $F(x)$ 有一个线性因子 $x-a$ $(a\in k)$，当有仅当 $F(a)=0$。

证明　如果 $x-a$ 是一个因子，则存在某个 $G(x)$ 使得 $F(x)=(x-a)G(x)$，进而有

$$F(a)=(a-a)G(a)=0\cdot G(a)$$

另一方面，假设 $F(a)=0$，利用除法算法可以得到 $F(x)=Q(x)\cdot(x-a)+R$，因为 $\deg R<\deg(x-a)=1$，则它必定是一个常数，两边代入 a 的值则得到

$$0=F(a)=Q(a)\cdot(a-a)+R=Q(a)\cdot 0+R=R$$

因此，$R=0$，因此 $x-a$ 整除 $F(x)$。 ♣

备注　这给出一种最为简单的验证线性因子的方法。

在 $\mathbf{Z}/2[x]$ 中只有一个次数为 0 的多项式，那就是常数1。零多项式的次数更适合说是无

穷。

在 $\mathbf{Z}/2[x]$ 中刚好有两个线性多项式，即 x 和 $x+1$。因为每个线性多项式是不可约的，所以它们是不可约的。

对于二次多项式，对线性项的系数和常数项各有两种选择，所以在 $\mathbf{Z}/2[x]$ 中有四个二次多项式。下面我们来考查它们的不可约性：

- 显然 $x^2 = x \cdot x$；
- 显然 $x^2 + x = x \cdot (x+1)$；
- $x^2 + 1 = (x+1)^2$，这里我们用到了 $2 = 0$ 这个事实，实际上
$$(x+1)^2 = x^2 + 2x + 1 = x^2 + 0 + 1 = x^2 + 1$$
- $x^2 + x + 1$：我们容易验证这个多项式是否有线性因子，$0^2 + 0 + 1 = 1 \neq 0$，$1^2 + 1 + 1 = 1 \neq 0$，所以 $x^2 + x + 1$ 在 $\mathbf{Z}/2[x]$ 上不可约。它也是 $\mathbf{Z}/2[x]$ 上惟一的二次不可约多项式。

对于系数在 $\mathbf{Z}/2$ 中的三次多项式，因为二次项系数、线性项系数和常数项分别有 2 种选择，所以总共有 8 个 $\mathbf{Z}/2$ 上的三次多项式。如果我们只考虑不可约的，则首先可以把常数项为 0 的（因为它们必定有线性因子 x）排除。对于那些非零系数且输入 1 得到 0 值的多项式，它们必定有一个线性因子 $x+1$。请记住，如果一个三次多项式可约，则它至少有一个线性因子。

因此在 $\mathbf{Z}/2[x]$ 上只有两个不可约的三次多项式，它们是 $x^3 + x^2 + 1$ 和 $x^3 + x + 1$。

在 $\mathbf{Z}/2[x]$ 中，三次项系数、二次项系数、线性项系数和常数项共有 $2^4 = 16$ 种选择。如果常数项为 0 或者非零系数的个数为偶数，则它必有线性因子 x 或 $x+1$。

这样就只有四个可能的四次不可约多项式：

$$x^4 + x^3 + x^2 + x + 1$$
$$x^4 + x^3 + 1$$
$$x^4 + x^2 + 1$$
$$x^4 + x + 1$$

容易看出这四个多项式在 $\mathbf{Z}/2[x]$ 中均没有线性因子。接下来我们寻找它们的（不可约的）二次因子，由前面的讨论已知，$\mathbf{Z}/2[x]$ 上的二次不可约多项式为 $x^2 + x + 1$，这样就只有 $x^4 + x^2 + 1 = (x^2 + x + 1)^2$ 是一个四次可约多项式。余下的三个四次多项式 $x^4 + x^3 + x^2 + x + 1$，$x^4 + x^3 + 1$ 和 $x^4 + x + 1$ 则为 $\mathbf{Z}/2[x]$ 上的四次不可约多项式。

$\mathbf{Z}/2[x]$ 中的五次多项式共有 32 个。除去那些常数项为 0 的就只剩下 16 个，再排除那些有偶数个非零系数项的多项式（它们有线性因子 $x+1$），就剩下 $\binom{4}{3} + \binom{4}{1} = 8$ 个需要进一步判别。这 8 个多项式没有线性因子，所以它们只可能是如下形式的低次多项式的乘积：

不可约的二次多项式 × 不可约的三次多项式

比如，如果存在两个二次不可约因子，则必定有一个线性因子。我们已经知道在 $\mathbf{Z}/2[x]$ 中只有一个二次不可约多项式，有两个三次不可约多项式。这样就有两个没有线性因子的五次可约多项式，它们是

$$(x^2 + x + 1) \cdot (x^3 + x^2 + 1) = x^5 + x + 1$$

$$(x^2+x+1)\cdot(x^3+x+1)=x^5+x^4+1$$

这样在 $\mathbf{Z}/2[x]$ 中就有六个五次不可约多项式。除了我们上面排除的可约的形式之外，那么它们必须是：具有常数项 1，非零系数的个数为奇数，当然还不能包括已经确认的两个可约的多项式 x^5+x^4+1 和 x^5+x+1。这样，有五个非零系数的四个五次不可约多项式分别是：

$$x^5+0+x^3+x^2+x+1$$
$$x^5+x^4+0+x^2+x+1$$
$$x^5+x^4+x^3+0+x+1$$
$$x^5+x^4+x^3+x^2+0+1$$

而有三个非零系数的两个五次不可约多项式是：x^5+x^3+1 和 x^5+x^2+1。

如果还有人幻想利用 $\mathbf{Z}/p[x]$ 中多项式分解的困难性来实现 RSA，那么请你赶紧放弃吧！因为这里已经有一种快速算法可以分解这些多项式了。

习题

19.4.01 求多项式环 $\mathbf{Q}[x]$ 中的多项式 $x^5+x^4+x^3+x^2+x+1$ 和 x^4+x^2+1 的最大公因子。

19.4.02 求多项式环 $\mathbf{Q}[x]$ 中的多项式 $x^{10}+x^8+x^6+x^4+x^2+1$ 和 $x^9+x^6+x^3+1$ 的最大公因子。

19.4.03 求多项式环 $\mathbf{F}_3[x]$ 中的多项式 x^6+x^3+1 和 x^2+x+1 的最大公因子，$\mathbf{F}_3$ 为包括三个元素的有限域。

19.4.04 求多项式环 $\mathbf{F}_2[x]$ 中的多项式 $x^6+x^4+x^2+1$ 和 $x^8+x^6+x^4+x^2+1$ 的最大公因子，$\mathbf{F}_2$ 为包含两个元素的有限域。

19.4.05 求多项式环 $\mathbf{F}_2[x]$ 中的多项式 x^5+x^4+1 和 x^5+x+1 的最大公因子，$\mathbf{F}_2$ 为包含两个元素有限域。

19.4.06 求系数在 $\mathbf{F}_2$ 中的两个多项式 $x^7+x^5+x^4+x^2+1$ 和 $x^8+x^7+x^4+x^3+1$ 的最大公因子。

19.4.07 求 $\mathbf{F}_2[x]$ 中的两个多项式 $x^{14}+x^{13}+x^{11}+x^{10}+x^9+x^7+x^5+x^4+x^3+1$ 和 $x^{14}+x^{12}+x^8+x^7+x^4+x^3+x+1$ 的最大公因子。

19.4.08 求系数在 $\mathbf{F}_2$ 中的两个多项式

$x^7+x^6+x^4+x^3+1$ 和 $x^8+x^7+x^5+x^3+x^2+x+1$ 的最大公因子。

19.4.09 将 $\mathbf{F}_2[x]$ 中的多项式 $x^{12}+x^{10}+x^7+x^6+1$ 分解为不可约因式的乘积。

19.4.10 将 $\mathbf{F}_2[x]$ 中的多项式 $x^{12}+x^9+x^8+x^7+x^6+x^5+1$ 分解为不可约因式的乘积。

19.4.11 将 $\mathbf{F}_2[x]$ 中的多项式 $x^{12}+x^{10}+x^9+x^8+x^7+x^5+x^4+x^3+1$ 分解为不可约因式的乘积。

19.5 欧几里得环

在介绍完**除法算法**和**欧几里得算法**以及域上重要的环的例子（如整数环和多项式环）之后，我们还要对这些结构进行抽象。目的是要证明**欧几里得环有惟一的分解**。这个结果也可用到整数上，只不过这里我们将证明以前想当然的结果，即整数有惟一的分解。

一个交换环 R 上的**绝对值**记为 $|r|$（$r\in R$）是一个具有如下性质函数：

- 可乘性：对所有的$r,s\in R$,$|rs|=|r|\cdot|s|$；
- 三角不等式：对所有的$r,s\in R$,$|r+s|\leqslant|r|+|s|$；
- 非负性：如果$|r|=0$，则$r=0$。

如果环R上的绝对值具有这样的性质，即对R的任何非空子集S，有一个最小正绝对值（S的元素的绝对值集合中）的元素，则称这个绝对值是**离散的**。

一个有单位元的交换环R是**欧几里得环**，如果存在一个R上的离散绝对值，记为$|r|$，那么对任意$x\in R$和任意$0\neq y\in R$，使得存在q，$r\in R$，也使得$x=yq+r$，且$|r|<|y|$。这就是说我们能够做作除法并得到一个比除式小的余式。

在上述定义中假设绝对值是离散的，这是相当重要的。一般情况下容易判断这个要求满足。比如，如果$|\ |$就是整数绝对值，也就成了通常的整数的绝对值，则良序原则就保证了“离散”性质。

欧几里得环重要的例子就是整数环$\mathbf{Z}$和域k上的多项式环$k[x]$，整数环$\mathbf{Z}$上的绝对值没什么好说，我们来看一下多项式环上的绝对值。定义$|P(x)|=2^{\text{degree}P}$且$|0|=0$。这里的2可以被其他任何大于1的数代替，这是一个多项式环上的绝对值。

- 欧几里得环R是整环。

证明 我们必须证明环R没有（非零）零因子，即必须证明如果$xy=0$，则x为0或者y为0。假设$xy=0$，则由范数的乘法性质得$0=|0|=|xy|=|x|\cdot|y|$。$|x|$和$|y|$都是非负的实数，而且它们的乘积为0，则必须其中之一为0，则由范数的非负性可得必有x,y之一为0。结论得证。♣

- 在一个欧几里得环R中，如果$r\in R$且$|r|<1$，则$r=0$。

证明 因为$|ab|=|a|\cdot|b|$，则有$|r^n|=|r|^n$。如果$r\neq0$，则方幂$|r|^n$形成了一个没有最小绝对值的值的集合，进一步它们构成了一个以0为极限但不包含0的递减序列。因此$r=0$。♣

- 在一个欧几里得环R中，一个元素$u\in R$是单位（即它有一个乘法逆元素），当且仅当$|u|=1$。特别地，$|1|=1$。

证明 因为$1=1\cdot1$，对此等式取绝对值并由绝对值的可乘性质，可得$|1|=|1\cdot1|=|1|\cdot|1|$。在实数中具有性质$z=z^2$只有0和1，因为$1\neq0$，我们得出$|1|\neq0$，则必有$|1|=1$。如果$uv=1$，取绝对值则有$1=|uv|=|u|\cdot|v|$。因为环上绝对值小于1的元素只有0（由前面的讨论），所以$|u|$和$|v|$都大于等于1。又因为它们的乘积为1，则它们必须都是1。因此单位的绝对值为1。另一方面，假设$|u|=1$，则运用除法算法，可以将1写为如下形式

$$1=q\cdot u+r$$

这里$|r|\lhd|u|$。因为$|u|=1$，所以$|r|<1$。由前面的结论可知，$r=0$。故$1=q\cdot u$，q就是u的乘法逆元素。因此任何绝对值为1的元素为单位。♣

定理 R为一欧几里得环，$x,y\in R$，形如$sx+ty$（$s,t\in R$）且有最小绝对值的元素为x,y的最大公因子。

证明 对绝对值离散性的假设，可以保证在绝对值$|sx+ty|$的非零值中有一个值是最小的。设$g=sx+ty$就是这样一个元素，我们必须证明$g\mid x$和$g\mid y$。利用除法/约简算法，我们有

$$x=q(sx+ty)+r$$

这里$|r|<|sx+ty|$。重新整理这个等式得到

$$r=(1-qs)x+(-qt)y$$

则r本身就是一个形如$s'x+t'y$（$s',t'\in R$）的元素。因为$sx+ty$是这类元素中绝对值最小的一个，而$|r|<|sx+ty|$，因此必有$r=0$。因此$sx+ty$整除x，类似地可得$sx+ty$整除y。这就证明了$sx+ty$是x和y的一个因子。另一方面，如果$d|x$，且$d|y$，则$d|(sx+ty)$，因此$sx+ty$能够被x和y的每个公因子整除。 ♣

命题 在欧几里得环R上，$p\in R$是一个素元，如果$d|p$，则有$|d|=1$或$|d|=|p|$。也就是说$r\in R$的一个真因子d满足$1<|d|<|r|$。

证明 我们首先来回忆一下交换环R上素元的定义：$p\in R$是一个素元，如果$ab=p$，则有a或者b为单位。为证明命题中的两个结论，只需要证明：如果$ab=n$，则a和b都不是单位当且仅当$1<|a|<|n|$。一方面，如果$1<|a|<|n|$，则因为$|n|=|ab|=|a|\cdot|b|$，可以得出$|n|>|b|>1$。又因为环R上的单位就是那些绝对值为 1 的元素，故a和b都不是单位。另一方面，如果$ab=n$且a和b都不是单位，则有$1<|a|$且$1<|b|$。又因为$|n|=|ab|=|a|\cdot|b|$，由此即得$|a|<|n|$且$|b|<|n|$。 ♣

引理 设p是欧几里得环R上的一个素元，如果$p|ab$，则$p|a$或$p|b$。更一般地，如果$p|a_1\cdots a_n$，则p必整除a_i其中之一。

证明 我们只需要证明如果$p|ab$且p不整除a，则$p|b$。因为p不整除a且p是一个素元，则p和a的最大公因子为 1。因此，存在$s,t\in R$，使得$1=sa+tp$。由此可得

$$b=b\cdot 1=b\cdot(sa+tp)=s(ab)+(bt)p$$

由于$p|ab$，当然有$p|b$。

一般地，如果$p|a_1\cdots a_n$，将乘积$a_1\cdots a_n$写为$(a_1)(a_2\cdots a_n)$。由第一部分结论，则有$p|a_1$或者$p|a_2\cdots a_n$。如果是第一种情况，则结论就已证明。如果是第二种情况，我们可以继续这一步骤，即将乘积$a_2\cdots a_n$写为$(a_2)(a_3\cdots a_n)$，则有$p|a_2$或者$p|a_3\cdots a_n$。重复这一步骤，则我们将证明p至少整除一个因子a_i。 ♣

定理 在欧几里得环R上，每个元素$r\in R$都可以分解为$r=up_1^{e_1}\cdots p_m^{e_m}$的形式，其中$u$是一个单位，$p_i$是不同的素元，$e_i$是正整数。如果$r=vq_1^{f_1}\cdots q_n^{f_n}$是另一个这样的分解，其中$v$是一个单位，$q_i$是素元，则$m=n$，并且可以重新排列顺序和编号使$q_i=p_i\times u_i$，对某些$u_i$和对所有的指标$i$。并且还有$e_i=f_i$。这也就是说这个分解是**惟一的**。

证明 首先我们来证明素因子分解的存在性。假设某些$r\in R$不存在一个素因子分解，那么利用离散性，存在一个$r\in R$没有素因子分解，且$|r|$是所有没有素因子分解的所有元素中最小的。如果r是一个素元，这当然它就是一个素因子分解。因此这个r不是一个素元，那么r就有一个真分解$r=ab$。由前面的结论，可以得到$1<|a|<|r|$和$1<|b|<|r|$。因为$|a|<|r|$和$|b|<|r|$，由r的极小性，则a和b都有素因子分解。那么r的素分解则可由a和b的素因子分解相乘而得到。（两个单位的乘积仍为一个单位!）

下面我们再来证明分解的惟一性，假设$r=u\cdot p_1^{e_1}\cdots p_m^{e_m}$且还有$r=v\cdot q_1^{f_1}\cdots q_n^{f_n}$，$p_i$和$q_i$均为素元。用归纳法，我们假设$m$是存在不同分解的最小整数，因为$p_1$整除$vq_1^{f_1}\cdots q_n^{f_n}$，而$p_1$又是素元，则由前面的引理，$p_1$必定整除$q_i$中的一个。因为可以重新编号，我们不妨设$p_1|q_1$。又因为这两个元素都是素元，则它们之间只差一个单位，也就是存在一个单位u_1，使$q_1=u_1\cdot p_1$。用u_1p_1代替q_1，可以得到

$$up_1^{e_1}\cdots p_m^{e_m}=vu_1^{f_1}\cdot p_1^{f_1}q_2^{f_2}q_3^{f_3}\cdots q_n^{f_n}$$

注意到 $vu_1^{f_1}$ 仍然是一个单位，因为 $e_1\geqslant 1$ 且 $f_1\geqslant 1$，这样就可以从等式两边消去至少一个 p_1。（我们已经证明欧几里得环是整环。）

由归纳法，因为我们已经假设 m 是出现在某个 $r\in R$ 的两个不同表示中出现的最小整数，在去除相同因子后余下的分解式就是相同的。在必要时利用单位调整素元后，分解式中的因子和因子的指数就相同了。 ♣

读者可能会注意到，在所有这些证明中我们并没有使用三角不等式。确实如此，但实际上根据公理化观点，任何作为绝对值的合理替代都建议是其自身，因为根据更实际的观点来看，它的功能和绝对值一样，包括三角不等式。

第20章 分圆多项式

现在我们来证明有关本原根的基本结论，它们是简单多项式 x^N-1 的因子性质的推论，这些结论在数学及其应用的许多地方都是非常重要的。

分圆多项式首次出现仅是作为一个辅助的研究对象，但对于帮助我们理解在 $\mathbf{Z}/m$ 和有限域上的代数机制方面是非常重要的。

20.1 特征

设 k 是一个域，k 的**特征** char k 就是使得

$$\underbrace{1_k+1_k+\cdots+1_k}_{n}=0_k$$

成立的最小正整数 n（如果存在），这里 1_k 是 k 的单位元，0_k 是其零元。通常我们简写为

$$\ell\cdot 1_k=\underbrace{1_k+1_k+\cdots+1_k}_{\ell}$$

ℓ 为正整数。

如果没有这样的正整数 n 存在，则我们称特征为 0。因此有 $\operatorname{char}\mathbf{Q}=0$，反之则为

$$\operatorname{char}(\mathbf{Z}/p)=p$$

命题 如果一个域的特征非零，则它为素数。对一特征为 p 的域，p 为素数，如果对某个正整数 n，有

$$\underbrace{1_k+1_k+\cdots+1_k}_{n}=0_k$$

则 $p\mid n$。

证明 假设

$$\underbrace{1_k+1_k+\cdots+1_k}_{n}=0_k$$

n 是满足此式的最小整数，并且 n 有一个分解

$$n=a\cdot b$$

a,b 是正整数。那么

$$\underbrace{(1_k+1_k+\cdots+1_k)}_{a}\cdot\underbrace{(1_k+1_k+\cdots+1_k)}_{b}=\underbrace{1_k+1_k+\cdots+1_k}_{n}=0_k$$

因为域中没有真的零因子，则必有 $a\cdot 1_k=0$ 或 $b\cdot 1_k=0$。我们假设 n 是最小的，如果 $a\cdot 1_k=0$，则 $a=n$，对 b 有类似的结论。因此，n 的分解 $n=a\cdot b$ 不是真分解。因为 n 没有真分解，则它是素数。

假设 $n\cdot 1_k=0_k$。根据除法算法，有 $n=qp+r$，$0\leqslant r<p$，则

$$0_k=n\cdot 1_k=q(p\cdot 1_k)+r\cdot 1_k=0_k+r\cdot 1_k$$

由此，$r\cdot 1_k=0_k$。因为 $r<p$，且 p 又是满足 $p\cdot 1_k=0_k$ 的最小正整数，则必有 $r=0$，即有 $p\mid n$。

正特征为 p 的域有一个不明显的特性，但这个特性（以下命题）却在理论和应用方面起着十分重要的作用。 ♣

命题 设 k 是特征为 p 的域，则对任何 $k[x]$ 上的多项式

$$f(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_2x^2+a_1x+a_0$$

有

$$f(x)^p=a_n^px^{pn}+a_{n-1}^px^{p(n-1)}+\cdots+a_2^px^{2p}+a_1^px^p+a_0^p$$

证明 回想一下，p 整除二项式系数 $\binom{p}{i}$，$0<i<p$。因此对 $0<i<p$

$$\binom{p}{i}\cdot 1_k=0\cdot k$$

那么对于 $a_n\in k$ 及任意系数在域 k 上的多项式 $g(x)$

$$(a_nx^n+g(x))^p=(a_nx^n)^p+\sum_{0<i<p}\binom{p}{i}(a_nx^n)^{p-i}g(x)^i+g(x)^p$$

因为中间项的系数

$$\binom{p}{i}\cdot 1_k=0_k$$

所以可以去掉，那么就剩下

$$(a_nx^n+g(x))^p=a_n^px^{pn}+g(x)^p$$

将这一结论运用于 $g(x)$ 上，取 $g(x)=a_{n-1}x^{n-1}+h(x)$

则有

$$g(x)=a_{n-1}^px^{p(n-1)}+h(x)^p$$

重复这一过程（也就是归纳法），可以得到 f 的结果

$$f(x)^p=a_n^px^{pn}+a_{n-1}^px^{p(n-1)}+\cdots+a_2^px^{2p}+a_1^px^p+a_0^p$$

这样就证明了结论。 ♣

例如，系数域 $k=\mathbf{Z}/p$，p 为素数，则有

$$(x+1)^p=x^p+\sum_{0<i<p}\binom{p}{i}x^i+1=x^p+1$$

并有

$$(x^2+1)^p=x^{2p}+1$$

$$(x^2+x+1)^p=x^{2p}+\quad{}^p+1$$

20.2 重因子

有一个简单的方法判断多项式中一个因子是不是多次出现（系数是域中元素）。这一方法在理论上和实际计算中都很有用处。设 k 为一个域，对于一个多项式

$$f(x)=c_nx^n+\cdots+c_1x+c_0$$

系数 c_i 属于 k，我们定义

$$f'(x)=nc_nx^{n-1}+(n-1)c_{n-1}x^{n-2}+\cdots+3c_3x^2+2c_2x+c_1$$

备注 注意我们只是简单地定义了一个“导数”，是纯代数意义上的，没有附加任何限制。当然这个公式已被假设产生了一些熟知的性质，比如乘法规则。我们可以利用自己的有关微分方面的知识做出“合理的猜测”。

引理 对多项式环 $k[x]$ 上的两个多项式 f,g，函数是域中元素，$r\in k$，有如下结论：

- $(r\cdot f)'=r\cdot f'$
- $(f+g)'=f'+g'$
- $(fg)'=f'g+fg'$

证明 第一个结论比较容易：令 $f(x)=a_mx^m+\cdots+a_0$，则计算

$$\begin{aligned}(r\cdot(a_mx^m+\cdots+a_0))'&=(ra_mx^m+ra_{m-1}x^{m-1}+\cdots+ra_0)'\\&=m\cdot(ra_m)x^{m-1}+(m-1)(ra_{m-1})x^{m-2}+\cdots+ra_1+0\\&=r(m\cdot(a_m)x^{m-1}+(m-1)\cdot(a_{m-1})x^{m-2}+\cdots+a_1+0)=r\cdot f'(x)\end{aligned}$$

证明第二个结论也不太困难：设 $f(x)=a_mx^m+\cdots+a_0$ 和 $g(x)=b_nx^n+\cdots+b_0$。因为可以给次数较低的一个多项式添加形如 $0\cdot x^\ell$ 的项，所以，不失一般性，可以假设 $m=n$，这样则有

$$\begin{aligned}&(f(x)+g(x))'=((a_n+b_n)x^n+\cdots+(a_1+b_1)x+(a_0+b_0)x^0)'\\&=n(a_n+b_n)x^{n-1}+(n-1)(a_{n-1}+b_{n-1})x^{n-2}+\cdots+1(a_1+b_1)x^0+0\cdot x^0\\&=(na_nx^{n-1}+(n-1)a_{n-1}x^{n-2}+\cdots+1\cdot a_1x^0)+(nb_nx^{n-1}+\cdots+(n-1)b_{n-1}x^{n-2}+\cdots+1\cdot b_1x^0)\\&=f'(x)+g'(x)\end{aligned}$$

现在来证明第三条性质（结论）。首先看一下如果 f 和 g 均是单项式，也就是 f，g 为简单的形式 $f(x)=ax^m$，$g(x)=bx^n$。一方面，我们有

$$(fg)'=(ax^m\cdot bx^n)'=(abx^{m+n})'=ab(m+n)x^{m+n-1}$$

另一方面，

$$f'g+fg'=amx^{m-1}\cdot bx^n+ax^m\cdot bnx^{n-1}=ab(m+n)x^{m+n-1}$$

这就证明了对于单项式 f 和 g 满足性质。

为了获得一般性的乘法规则，设

$$\begin{aligned}f(x)&=a_mx^m+\cdots+a_0\\g(x)&=b_nx^n+\cdots+b_0\end{aligned}$$

在乘积 $f(x)g(x)$ 中 x^ℓ 的系数为

$$\sum_{i+j=\ell}a_i\cdot b_j$$

在乘积的导数中 $x^{\ell-1}$ 的系数为

$$\ell\sum_{i+j=\ell}a_i\cdot b_j$$

另一方面，$f'g$ 中 $x^{\ell-1}$ 项的系数为

$$\sum_{i+j=\ell}(ia_i)\cdot b_j$$

fg' 中 $x^{\ell-1}$ 项的系数为

$$\sum_{i+j=\ell}a_i\cdot jb_j$$

将这两项加在一起，就得到 $f'g+fg'$ 中 $x^{\ell-1}$ 项系数为

$$\sum_{i+j=\ell} a_i \cdot b_j \cdot (i+j) = \ell \sum_{i+j=\ell} a_i \cdot b_j$$

它刚好就是 $(fg)'$ 中对应项的系数。这就证明了乘法规则。 ♣

命题 设 f 是系数为域 k 上的多项式，P 是系数为域 k 上的不可约多项式。一方面，如果 P^2 乘除 f，则 P 整除 $gcd(f,f')$。另一方面，如果 k 的特征 P 是正数，P 是可约的，每个元素 $a \in k$ 在 k 中有 P 次根，且 P 整除 $gcd(f,f')$，则有 P^2 整除 f。（注意最后一个条件一定成立，例如，在 P 为素数的有限域 $\mathbf{Z}/P$ 上。）

证明 一方面，假设 $f = P^2 \cdot g$，利用乘法规则有

$$f' = 2PP' \cdot g + P^2 \cdot g' = P \cdot (2P'g + Pg')$$

f' 一定是 P 的倍数。这一半的证明还没有用到 P 的不可约性。

另一方面，假设 P 整除 f 和 f'（实际上 P^2 整除 f）。用 P 除 f/P，可以得到

$$f/P = Q \cdot P + R$$

R 的次数小于 P 的次数，那么 $f = QP^2 + RP$。两边取导数则有

$$f' = Q'P^2 + 2QPP' + R'P + RP'$$

由假设 $P \mid f'$，则等式右边的所有项可能除了 RP' 均可被 P 整除，所以，P 亦整除 RP'。由于 P 是不可约的，它又整除 RP'，则必有 $P \mid R$ 或 $P \mid P'$。如果 $P \mid R$，而 R 的次数小于 P 的次数，所以 $R = 0$。即有 $P^2 \mid f$，这就证明了我们的结论。

如果 P 不整除 R，则必有 $P \mid P'$。因为 P' 的次数要低于 P，如果 P 整除 P'，则 P' 必为零多项式。我们来看一下这对不可约的 P 是不可能的。设

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$$

那么

$$P'(x) = n a_n x^{n-1} + (n-1) a_{n-1} x^{n-2} + \cdots + 2a_2 x^1 + a_1 + 0$$

这个多项式要为零多项式，必有

$$\ell \cdot a_\ell = 0$$

ℓ 为所有的下标。这也就是说，对任何下标 ℓ，$a_\ell \neq 0$，则必有 $\ell \cdot 1_k = 0_k$。因为 P 至少有一个系数是非零的，这就是说 k 的特征不为 0，它必为某个素数 p。由前面的 $\ell \cdot 1_k = 0_k$，就可以得到 $p \mid \ell$。这就可以得出：如果系数 a_ℓ 非零，则特征 $p \mid \ell$，所以

$$f(x) = a_{pm} x^{pn} + a_{p(m-1)} x^{p(m-1)} + a_{p(m-2)} x^{p(m-2)} + \cdots + a_{2p} x^{2p} + a_p x^p + a_0$$

令 b_i 是 k 中 a_i 的 p 次方根。由前面的结论，我们认识到 $f(x)$ 是如下多项式的 p 次方幂，

$$b_{pm} x^n + b_{p(m-1)} x^{(m-1)} + b_{p(m-2)} x^{(m-2)} + \cdots + b_{2p} x^2 + b_p x + b_0$$

但如果 P 是个 p 次方幂，它当然不是不可约的。因此，对 P 不可约的情况，不可能有 P' 为零多项式。因此，只可能有 $R = 0$。于是 P^2 整除 f 的结论得证。 ♣

习题

20.2.01 $\mathbf{F}_2[x]$ 上的多项式 $x^4 + x^2 + 1$ 有一个重因子，请求出这个重因子。

20.2.02 $\mathbf{F}_2[x]$ 上的多项式 $x^6 + x^4 + x^2 + 1$ 有一个重因子，请求出这个重因子。

20.2.03 $\mathbf{F}_2[x]$ 上的多项式 $x^8 + x^7 + x^6 + x^4 + x^3 + x + 1$ 有一个重因子，请求出这个重因子。

20.2.04 $\mathbf{F}_2[x]$ 上的多项式 $x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$ 有一个重因子，请求出这个重因子。

20.2.05 $\mathbf{F}_2[x]$ 上的多项式 $x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ 有一个重因子，请求出这个重因子。

20.3 解分圆多项式

k 是一个域，$b \in k$，则 b 的指数就是满足 $b^n = 1$ 的最小正整数 n（如果存在的话）。即 $b^n = 1$，但对 $0 < d < n$，$b^d \neq 1$。换句话说，b 是多项式 $x^n - 1$ 的根，但 b 不是 $x^d - 1$ 的根。我们将要描述的多项式 φ_n 是 n 次分圆多项式，b 是其一个指数为 n 的根。

给定域 k，n 是一个不被 k 的特征整除的整数。（如果特征为 0，则这根本就不是条件。）

引理 m, n 为两个整数（可以被特征整除或不整除），则有

$$gcd(x^m - 1, x^n - 1) = x^{gcd(m,n)} - 1$$

$$lcm(x^m - 1, x^n - 1) = x^{lcm(m,n)} - 1$$

证明 我们对 m 和 n 的最大值做归纳法。首先，如果 $m = n$，$x^m - 1 = x^n - 1$，那么结论是显然的。其次，如果 $m > n$，做一个因子的变形

$$x^m - 1 - x^{m-n} \cdot (x^n - 1) = x^{m-n} - 1$$

因为如果 D 是一个多项式，且 D 能整除 $x^m - 1$ 和 $x^n - 1$，则 D 也整除 $x^{m-n} - 1$。由归纳法

$$gcd\,(x^{m-n} - 1, x^n - 1) = x^{gcd\,(m-n,n)} - 1$$

但是

$$gcd\,(m, n) = gcd\,(m - n, n)$$

且

$$x^m - 1 = x^{m-n} \cdot (x^n - 1) + x^{m-n} - 1$$

所以

$$gcd\,(x^m - 1, x^n - 1) = gcd\,(x^{m-n} - 1, x^n - 1)$$

再次，如果 $m < n$，我们将 m, n 互换，重复上述过程，做一个因子的变形。

$$x^n - 1 - x^{n-m} \cdot (x^m - 1) = x^{n-m} - 1$$

因此如果 D 是能整除 $x^m - 1$ 和 $x^n - 1$ 的多项式，则 D 整除 $x^{n-m} - 1$。由归纳法

$$gcd\,(x^{n-m} - 1, x^n - 1) = x^{gcd\,(n-m,m)} - 1$$

但是

$$gcd(m, n) = gcd(n - m, n)$$

且

$$x^n - 1 = x^{n-m} \cdot (x^m - 1) + x^{n-m} - 1$$

所以

$$gcd\,(x^m - 1, x^n - 1) = gcd\,(x^{n-m} - 1, x^m - 1)$$

这就完成了归纳步骤。最小公倍因子的证明与此情形完全类似。 ♣

引理 设 k 为一个域，n 是一个正整数，k 的特征不整除 n，则多项式 $x^n - 1$ 没有重因子。

证明 由前面的结论，只需要验证 $x^n - 1$ 和其导数 nx^{n-1} 的最大公因子 gcd 为 1 即可。因为域 k 的特征不整除 n，则 $n \cdot 1_k$ 有一个乘法逆元 t，$t \in k$。我们来做一个带余除法：

$$(x^n - 1) - (tx) \cdot (nx^{n-1}) = -1$$

因此，$gcd(x^n - 1, nx^{n-1}) = 1$。引理得证。 ♣

现在假设 n 不被域 k 的特征整除，定义 n 次**分圆多项式** $\varphi_n(x)$（系数在 k 中）为

$$\varphi_n(x)=\frac{x^n-1}{lcm\{\{x^d-1:0<d<n,d\mid n\}\}}$$

这里所取的最小公倍数为首一的。

定理 设 m,n 为整数，域 k 的特征不整除 m,n，则

- φ_n 是首一的；
- $gcd(\varphi_m,\varphi_n)=1$；
- φ_n 的次数等于 $\varphi(n)$（欧拉函数）；
- $\varphi_n(x)$ 还有一个更为有效的表示为 $\varphi_n(x)=\dfrac{x^n-1}{\prod\limits_{1\leqslant d<n,d\mid n}\varphi_d(x)}$；
- 多项式 x^n-1 可分解为 $x^n-1=\prod\limits_{1\leqslant d\leqslant n,d\mid n}\varphi_d(x)$；

证明 首先，我们应当验证 x^d-1，$d<n$，$d\mid n$ 的最小公倍数整除 x^n-1，由于 $d\mid n$（且 $d>0$），则有 $x^d-1\mid x^n-1$（这可由高中代数或前面的引理得到）。因此，利用系数为域上的多项式的惟一分解，可得所有能整除 x^n-1 的因式的最小公倍 lcm 也能整除 x^n-1。

其次，φ_n 为首一的结论可由 φ_n 的定义得到，因为它是首一多项式 x^n-1 与首一的最小公倍的商，必然是首一的。

再次，我们来确定 $gcd(\varphi_m,\varphi_n)$，首先注意到 $\varphi_m\mid x^m-1$，$\varphi_n\mid x^n-1$。因此，

$$gcd(\varphi_m,\varphi_n)\text{ 整除 }gcd(x^m-1,x^n-1)$$

由前面的引理，我们可以得到

$$gcd\ (x^m-1,x^n-1)=x^{gcd\ (m,n)}-1$$

再由 φ_m 的定义，φ_m 整除

$$\frac{x^m-1}{x^{gcd\ (m,n)}-1}$$

所以 $gcd(\varphi_m,\varphi_n)$ 亦整除上式。因为 n 不被特征整除，前面的引理已表明 x^n-1 没有重因子。因此，由 $gcd(\varphi_n,\varphi_m)$ 整除 $x^{gcd(m,n)}-1$ 的事实以及 $(x^n-1)/(x^{gcd(m,n)}-1)$，可以得到 $gcd\ (x^m-1,x^n-1)=1$。

第四，我们用归纳法来证明

$$x^n-1=\prod_{1\leqslant d\leqslant n,d\mid n}\varphi_d(x)$$

对 $n=1$，结论正确，由 φ_n 的定义，

$$x^n-1=\varphi_n(x)\cdot lcm\{x^d-1:d\mid n,0<d<n\}$$

对 $d<n$，

$$x^d-1=\prod_{0<e<d,e\mid d}\varphi_e(x)$$

因为我们已经证明，对 $m\neq n$，有 $gcd(\varphi_m,\varphi_n)=1$，故有

$$lcm\{x^d-1:d\mid n,0<d<n\}=\prod_{d\mid n.d<n}\varphi_d(x)$$

因此

$$x^n-1=\varphi_n(x)\cdot\prod_{d|n,d<n}\varphi_d(x)$$

关于 φ_n 阶数的结论可以由已经证明的欧拉函数等式 $\sum_{d|n,d>0}\varphi(d)=n$ 来推得。

至此，定理得证。 ♣

习题

20.3.01 求分圆多项式 φ_2、φ_3、φ_4、φ_5、φ_6、φ_8、φ_9、φ_{12}。

20.3.02 不用直接由定义计算的方法，求分圆多项式 φ_{14}、φ_{16}、φ_{18}、φ_{20}、φ_{24}、φ_{25}。

20.3.03 用两种方法求分圆多项式 φ_{15}、φ_{21}。

20.3.04(*) 求一个系数不为 0、+1、−1 的分圆多项式。

20.4 本原根

现在我们来证明任何有限域 k 的乘法群 $k^\times$ 为*循环群*。$k^\times$ 的生成元有时称为 k 的*本原根*。$k^\times$ 的这个性质对现代的素性检验和现代因式分解算法非常关键。

定理 设 k 为一有限域，则 $k^\times$ 为一循环群。

证明 设 q 是 k 中元素的个数，单位群 $k^\times$ 是一个群。因为 k 是一个域，则任何 $0\neq b\in k$ 都在 k 中有逆元。因此 $k^\times$ 的阶为 $q-1$。因此，由拉格朗日定理的推论，对 $b\neq 0$，

$$b^{q-1}=1$$

也就是说，k 中的任何非零元素均是多项式 $f(x)=x^{q-1}-1$ 的一个根。另一方面，由代数基本定理，这个多项式至多在 k 中有 $q-1$ 个根。因此，$f(x)$ 在 k 中恰有 $q-1$ 个（不同）的根。

设 p 为 k 的特征，p 当然不能整除 $q-1$。因为如果 $p|q-1$，则 $f(x)=x^{q-1}-1$ 的导数将会是 0。因此，$gcd(f,f')=f$，那么 f 就会有重根，而前面已指出 f 恰有 $q-1$ 个不同的根，所以这不可能。

因为 k 不整除 $q-1$，我们还可以运用前面一节有关分圆多项式的结果。即

$$x^{q-1}-1=\prod_{d|q-1}\varphi_d(x)$$

由于 $x^{q-1}-1$ 在 k 中有 $q-1$ 个不同的根，并且因为这里的 φ_d 两两互素，那么对 $d|q-1$ 的 φ_d，它的阶数等于它在 k 中根的个数。因此 φ_d 对 $d|q-1$ 在 k 中就有 $\varphi(d)>0$ 个根(欧拉函数)。

最后，$\varphi_{q-1}(x)$ 的根就是那些满足 $b^{q-1}=1$，且不存在 $e<q-1$ 使得 $b^e=1$ 的域元素。本原根就是 $\varphi_{q-1}(x)$ 的根。分圆多项式 φ_{q-1} 有 $\varphi(q-1)$ 个根。因此，就存在 $\varphi(q-1)>0$ 个本原根。即群 $k^\times$ 有一个生成元，也就是说 $k^\times$ 为循环群。 ♣

20.5 模 p 的本原根

现在我们来验证 p 个元素的有限域 $\mathbf{Z}/p$ 的乘法群 $\mathbf{Z}/p^\times$ 为一个*循环群*。该群的任何生成元称为 $\mathbf{Z}/p$ 的*本原根*。$\mathbf{Z}/p$ 的这一性质在素性检验和因式分解算法中相当关键。

定理 p 是素数，单位群 $\mathbf{Z}/p^\times$ 为一循环群。

证明 作为研究分圆多项式的推论，我们已经证明了任何有限域 k 的乘法群 $k^\times$ 都是循环

群。因此，我们需要做的就是验证 $\mathbf{Z}/p$ 是域，也就是说必须验证任何非零元素 $b\in\mathbf{Z}/p$ 均有一个乘法逆元。

我们再来解释一下，为什么存在乘法逆元。因为在以前的内容中已经论述过了，实际上，p 为素数，如果 $b\neq 0 \bmod p$，则 $gcd(p,b)=1$。因此存在整数 s,t，使得 $sp+tb=1$，对这个等式模 p，则有 $tb=1 \bmod p$。也即 t 为 $b \bmod p$ 的乘法逆元。 ♣

习题

20.5.01 求 mod11 和 mod13 的本原根。

20.5.02 求 mod17 的本原根。

20.5.03 说明为什么容易验证 2 不是模 7、17、31 的本原根。

20.5.04 证明 2 和 3 都不是模 23 的本原根，但 5 却是模 23 的本原根。

20.5.05 证明 2、3、4、5 都不是模 41 的本原根，但 6 是模 41 的本原根。

20.5.06 证明模 191 没有小于 19（整数）的本原根，但 19 却是一个本原根。

20.6 素数方幂

因为我们已经知道，对 $\mathbf{Z}/p$（p 为奇素数)存在本原根，那么证明 $\mathbf{Z}/p^e$ 中存在本原根就不难了。对于 $\mathbf{Z}/2p^e$ 也是同样道理。

定理 对于奇素数 p，$\mathbf{Z}/p^e$ 和 $\mathbf{Z}/2p^e$ 都有本原根，也就是说乘法群 $\mathbf{Z}/p^{e\times}$ 和 $\mathbf{Z}/2p^{e\times}$ 都是循环群。

推论 对于模 p 的一个本原根整数 g，g 或者（$1+p$）g 是 $\bmod p^e$ 和 $\bmod 2p^e$（$e\geqslant 1$）的本原根。特别的，如果 $g^{p-1}\neq 1 \bmod p^2$，则 g 是 $\bmod p^e$ 和 $\bmod 2p^e$（对所有 $e\geqslant 1$）的本原根，否则 $(1+p)g$ 会取而代之。

在证明定理和推论之前，我们来看下面这个命题。它的要点就是掌握 $\mathbf{Z}/p^{e\times}$ 中一定类型元素的阶，比我们去证明 $\mathbf{Z}/p$ 有一个本原根更加重要。这个命题可用来证明关于本原根的定理和推论。

命题 设 p 是奇素数，对于 $1\leqslant k\leqslant e$ 的整数 k，以及整数 x，p 不整除 x，则 $\mathbf{Z}/p^{e\times}$ 中元素 $1+p^k x$ 的阶为 p^{e-k}。特别的，对 p 不整除 x 且 $k\geqslant 1$，

$$(1+p^k x)^{p^\ell}=1+p^{k+\ell}y,\qquad y=x \bmod p$$

证明 这里的一个重要技巧就是素数 p 整除二项式系数

$$\binom{p}{1},\binom{p}{2},\cdots,\binom{p}{p-2},\binom{p}{p-1}$$

而且 p 是奇素数，所以假设 $p>2$，首先来计算

$$(1+p^k x)^p=1+\binom{p}{1}p^k x+\binom{p}{2}p^{2k}x^2+\cdots+\binom{p}{p-1}p^{(p-1)k}x^{p-1}+p^{pk}x^p$$

$$=1+p^{k+1}\cdot\underbrace{(x+\binom{p}{2}p^{2k-(k+1)}x^2+\cdots+p^{pk-(k+1)}x^p)}_{y}$$

因为 p 整除二项式系数，所以表达式 y 模 p 后同余 x。我们来看最后一项 $p^{pk-(k+1)}x^p$，由于我们知道 $k\geqslant 1$ 且 $p>2$，所以 $pk-(k+1)\geqslant 1$。这样我们就证明了

$$(1+p^k x)^p = 1+p^{k+1}y \quad , \qquad y = x \bmod p$$

重复利用这一结果（即做归纳法），可得

$$(1+p^k x)^{p^\ell} = 1+p^{k+\ell}y \quad , \qquad y = x \bmod p$$

这就是命题中的那个式子。

现在我们来证明关于阶的论断。首先我们来看一下形如 $1+px$ 的元素在 $\mathbf{Z}/p^{e\times}$ 中的阶是怎样的。这里我们要运用拉格朗日定理。因此可以对 $\mathbf{Z}/p^{e\times}$ 中形如 $1+px$ 的元素进行计数。首先，对任意整数 x，整数 $1+px$ 与 p 互素，这就得到了 $\mathbf{Z}/p^{e\times}$ 的一个元素。另一方面，如果

$$1+px = 1+px' \bmod p^e$$

那么 $p^e \mid (1+px-1-px')$。即 $p^{e-1} \mid (x-x')$。所以如果 $x = x' \bmod p^{e-1}$，则整数 $1+px$ 和 $1+px'$ 表示 $\mathbf{Z}/p^{e\times}$ 中的同一个元素。同时，p^{e-1} 个整数 $x = 0,1,2,\cdots,p^{e-1}-1$ 给出了 $\mathbf{Z}/p^{e\times}$ 中所有表示为 $1+px$ 的元素。

由拉格朗日定理，$\mathbf{Z}/p^{e\times}$ 中任何形如 $1+px$ 的元素的阶必整除 p^{e-1}。

这个限制可以让我们计算 $(1+p^k x)^{p^\ell}$，来给出阶这个问题的确定性答案：对于 p 不整除 x 的情况：

$$(1+p^k x)^{p^\ell} = 1+p^{k+\ell}y$$

这里 $y = x \bmod p$。因此除非 $k+\ell \geqslant e$，否则上式模 p^e 不为 1。因此 $1+p^k x \bmod p^e$ 的阶为 p^{e-k}，这就证明了命题。 ♣

推论的证明 因为推论的结论比定理结论更强，因此只需要证明推论中的论断即可得到定理的证明。

在开始我们的主要工作之前，我们来看一下为什么 $\mathbf{Z}/p^e$ 的一个本原根 g 也是 $\mathbf{Z}/2p^{e\times}$ 的一个本原根。主要的理由是，对于奇素数 p

$$\varphi(2p^e) = (2-1)(p-1)p^{e-1} = (p-1)p^{e-1} = \varphi(p^e)$$

令 g 是 $\bmod p^e$ 的一个本原根，则 $\ell = \varphi(p^e)$ 是满足 $g^\ell = 1 \bmod p^e$ 的最小整数。因此，就不存在更小的指数 ℓ，使得 $g^\ell = 1 \bmod 2p^e$。又因为 $p^e \mid 2p^e$，所以一个 $\bmod p^e$ 的本原根也是 $\bmod 2p^e$ 的本原根。

现在回到中心问题上来，即 $\mathbf{Z}/p^e$ 的本原根。我们要证明的就是乘法群 $\mathbf{Z}/p^{e\times}$ 对某些 g 具有形式$\langle g\rangle$。令 g_1 是模 p 的一个本原根（存在性是有保证的），现在需要对 g_1 做一些适当的“校正”以得到一个模 p^e 的本原根，这有点像亨泽尔引理的方法。事实表明最多只需要一个简单的校正即可，所以从某个角度来说，这要比亨泽尔引理中的运用简单些。

如果 $g_1^{p-1} = 1+px$，且 p 不整除 x，那么我们来证明 g_1 是 $\bmod p^e$ 的一个本原根，$e \geqslant 1$。由拉格朗日定理，g_1 在 $\mathbf{Z}/p^{e\times}$ 中的阶为 $\varphi(p^e) = (p-1)p^{e-1}$ 的因子。因为 $p-1$ 是满足 $g_1^\ell = 1 \bmod p$ 的最小正指数 ℓ，所以 $p-1$ 整除 g_1 在 $\mathbf{Z}/p^{e\times}$ 中的阶。（由循环子群的讨论即知）因此，g_1 的阶必是下列数之一

$$p-1,(p-1)p,(p-1)p^2,\cdots,(p-1)p^{e-1}$$

因此，问题转化为寻找最小的整数 ℓ，使得

$$g_1^{(p-1)p^\ell} = 1 \bmod p^e$$

而我们已假设 $g_1^{p-1} = 1+px$，p 不整除 x，那么就是要寻找最小的正整数 ℓ，满足

$$(1+px)^{p^\ell} = 1 \bmod p^e$$

由命题的结论，满足这一性质的最小正整数 ℓ 为 $e-1$。这也就是说我们已经证明对所有 $e\geqslant 1$，g_1 是模 p^e 的一个本原根。

现在我们假设

$$g_1^{p-1}=1+px$$

其中 $p\mid x$。考虑

$$g=(1+p)g_1$$

显然 g 仍是模 p 的本原根。因为 $g=g_1 \bmod p$。下面计算

$$(1+p)^{p-1}=1+\binom{p-1}{1}p+\binom{p-1}{2}p^2+\cdots+\binom{p-1}{p-2}p^{p-2}+p^{p-1}$$

$$1+p\cdot\underbrace{\left(\binom{p-1}{1}+\binom{p-1}{2}p+\binom{p-1}{3}p^2+\cdots\right)}_{y}=1+py$$

由于

$$\binom{p-1}{1}=p-1$$

所以

$$y=p-1 \bmod p$$

因此 p 不整除 y，所以

$$g^{p-1}=((1+p)g_1)^{p-1}=(1+py)(1+px)=1+p(y+x+pxy)$$

因为 $p\mid x$，所以

$$y+x+pxy=y \bmod p$$

特别的，p 不整除 $y+x+pxy$。因此，通过对本原根的一些校正，我们又返回到前面一种情况，即 g^{p-1} 具有 $g^{p-1}=1+pz$，p 不整除 z 的形式。这种情形已经证明这个 g 是模 p^e，$e\geqslant 1$ 的一个本原根。

这就完成了对在 p 为奇素数时，$\mathbf{Z}/p^e$ 中本原根的存在性的证明。 ♣

20.7 本原根的计数

在证明了本原根的存在性之后，我们感兴趣的就是本原根到底有多少。

定理 如果 $\mathbf{Z}/n$ 有一个本原根，则存在

$$\varphi(\varphi(n))$$

个模 n 的本原根。（$\varphi(n)$ 是欧拉函数）。例如，若 p 奇素数，那么就有

$$\varphi(\varphi(p^e))=\varphi(p-1)\cdot(p-1)p^{e-2}$$

个模 p^e 的本原根。

证明 $\mathbf{Z}/n$ 有本原根的假设说明乘法群 $\mathbf{Z}/n^\times$ 是循环群。则对某个本原根 g

$$\mathbf{Z}/n^\times=\langle g\rangle$$

当然，g 的阶 $|g|$ 必是 $\mathbf{Z}/n^\times$ 的阶 $\varphi(n)$。那么从循环群的讨论，我们可以列出 $\langle g\rangle$ 的所有不同元素：

$$g^0,g^1,g^2,g^3,\cdots,g^{\varphi(n)-1}$$

并且

$$g^k \text{的阶} = |g^k| = \frac{|g|\text{的阶}}{gcd(k,|g|)}$$

所以$\langle g\rangle$的生成元必须是满足如下条件的元素：

$$g^k,\ 1 \leqslant k \leqslant |g| \text{且} k \text{与} |g| \text{互素}$$

由欧拉函数的定义，就有$\varphi(|g|)$个这样的元素。因此，由$|g| = \varphi(n)$，就是说有$\varphi(\varphi(n))$个这样的元素，即有$\varphi(\varphi(n))$个本原根。 ♣

推论 对奇素数p，$\mathbf{Z}/p^{ex}$中本原元的比例为$\varphi(p-1)/p$。

证明 这只需要将本原根的数量与$\mathbf{Z}/p^{ex}$中的元素数量做比较即可知

$$\frac{\varphi(\varphi(p^e))}{\varphi(p^e)} = \frac{\varphi(p-1)\cdot(p-1)p^{e-2}}{(p-1)p^{e-1}} = \frac{\varphi(p-1)}{p}$$

推论得证。 ♣

备注 可见模p^e的本原根还是相当多的。

20.8 不存在性

对一般的整数n，$\mathbf{Z}/n$中不存在本原根。

定理 如果n不为2，4，也不具有p^e，$2p^e$这样的形式，对奇素数p（e是正整数），则不存在模n的本原根。

证明 首先，我们来看一下$\mathbf{Z}/2^e$，$e \geqslant 3$。任何元素$b \in \mathbf{Z}/2^{ex}$都能表示为$b = 1+2x$的形式，对某个整数x，有

$$(1+2x)^2 = 1+4x+4x^2 = 1+4x(x+1)$$

这里有一个特别之处就是对任意整数x，$x(x+1)$都能被2整除。实际上，如果x是偶数，自然$x(x+1)$为偶数；如果x是奇数，则$x(x+1)$亦为偶数。因此

$$(1+2x)^2 = 1 \bmod 8$$

由如下表示 $(1+2^k x)^2 = 1+2^{k+1}x+2^{2k}x^2$，可用归纳法证明

$$(1+8x)^{2^{e-3}} = 1 \bmod 2^e$$

两者结合起来，便有

$$(1+2x)^{2^{e-2}} = 1 \bmod 2^e$$

但是$2^{e-2} < 2^{e-1} = \varphi(2^e)$，即对$e>2$，不存在模$2^e$的本原根。

现在我们考虑n不为2的方幂的情形。对n可表为$n = p^e m$，p是奇素数，且不整除m，由欧拉定理，有

$$b^{\varphi(p^e)} = 1 \bmod p^e$$
$$b^{\varphi(m)} = 1 \bmod m$$

令$M = lcm(\varphi(p^e),\varphi(m))$，则

$$b^M = (b^{\varphi(p^e)})^{M/\varphi(p^e)} = 1^{M/\varphi(p^e)} = 1 \bmod p^e$$

且

$$b^M = (b^{\varphi(m)})^{M/\varphi(m)} = 1^{M/\varphi(m)} = 1 \bmod m$$

因此有

$$b^M = 1 \bmod p^e m$$

但是一个本原根g有这样的性质，没有比$\varphi(p^e m)$更小的指数ℓ，使得$g^\ell = 1 \bmod p^e m$。因此，

除非

$$gcd(\varphi(p^e),\varphi(m))=1$$

否则

$$lcm(\varphi(p^e),\varphi(m))<\varphi(p^e)\varphi(m)=\varphi(p^e m)$$

这就否定了存在一个本原根的可能性。

这里，我们需要 $\varphi(m)$ 与 $\varphi(p^e)=(p-1)p^{e-1}$ 互素。实际上这是满足的。因为 $p-1$ 是偶数，这意味着 $\varphi(m)$ 必须是奇数。如果一个奇素数 $q\mid m$，则 $q-1$ 整除 $\varphi(m)$，这就必使 $\varphi(m)$ 为偶数，这是不可能的。因此，没有奇素数整除 m。更进一步，如果比2大的2的方幂整除 m，必有 m 为偶数，所以不存在本原根。

因此，除了我们已证明的情况存在本原根外，不存在模 n 的本原根。 ♣

20.9 搜索算法

p 为素数，如果我们知道 $p-1$ 的因子分解，则存在一个合理的算法来寻找模 p 的本原根，因为我们有一个有效的标准来检验一个元素 b 是否为模 p 的本原根。

引理 p 为素数，整数 b 为模 p 的本原根，当且仅当对所有能整除 $p-1$ 的素数 q，有

$$b^{(p-1)/q}\neq 1 \bmod p$$

证明 如果 b 是一个本原根，引理的结论显然是满足的。另一方面，假设对一特定的元素 b，它满足引理的条件。令 q^e 为能整除 $p-1$ 的素数 q 的方幂，并令 t 是 q 在 $\mathbf{Z}/p^{\times}$ 中的阶，则由费马小定理，$t\mid p-1$。如果 q^e 不能整除 t，则有 $t\mid(p-1)/q$。但是由假设，t 不能整除 $(p-1)/q$，因此 $q^e\mid t$。因为这对任何整除 $p-1$ 的素数 q 都成立，由整数的惟一分解，最小公倍数 m 也整除 t。当然，所有这些能整除 $p-1$ 的素数方幂的最小公倍数就是这个数本身。即 $m=p-1$，且 $p-1\mid t$。因为 $t\mid p-1$，故 $t=p-1$。这样，b 就是模 p 的一个本原根。 ♣

备注 注意能整除 $p-1$ 的素数的个数小于 $\log_2 p$。

备注 回想一下，模 p 本原根的数量为 $\varphi(p-1)$，这个数量大于 $(p-1)/4$。因此随机选择本原根命中的可能至少为1/4。因此一般经过2或3次试验后，应当能找到一个本原根。

搜索模 p（p 为素数）本原根的算法亦可描述如下（利用 $p-1$ 的分解知识）：

- 随机选择一个 b。
- 对于能整除 $p-1$ 的每个素数 q，计算 $b^{(p-1)/q} \bmod p$。
- 如果这些值有任何一个 $\bmod p$ 等于1，则放弃 b 并选择另一个元素。
- 如果这些值没有一个等于1$(\bmod p)$，则 b 就是一个模 p 本原根。

备注 因为在 $\mathbf{Z}/p^{\times}$ 中至少有1/4的元素为本原根。那么在寻找一个本原根时，随机选择的成功率也较高。上述引理给出了判断一个元素是否为本原根的有效方法。

备注 通常，2或3为一本原根，在1000以内的168个素数中，仅有60个素数，2或3不是它们的本原根。而且在这168个素数中，也仅有7个模数191，311，409，439，457，479和911的本原根不在2，3，5，6，7，10，11中出现。

第21章 随机数发生器

任何由确定的过程生成的随机数序列，也就是输出完全由输入所决定的序列，从实际意义上说都不是随机的。但我们仍想知道“随机数”到底是什么含义。要想准确地描述这个概念是比较困难的。

为了获得“真随机数”序列，最好的来源似乎是放射性物质。普遍且经常使用的随机源，如**键盘反应时间**（即两次击键之间的延迟），往往具有较好的结构，以至于它们不适用于较重要且需要大量随机数的密码应用。

由于我们不能生成真正随机的数，所以我们就使用**伪随机数发生器**（pseudo-random number generators），或 **pRNG**。

我们对pRNG感兴趣的一个应用是产生一个一次一密乱码本，这将在下一节中进行讨论。但是要想获得用于此目的的质量很好的 pRNG 是很困难的：关键问题是一个攻击者能否根据密钥流的已知位去预测后面的位。特别的，我们将要讨论的线性移位寄存器（LFSR）是很确定的而且质量不够好，因为 Massey-Berlekamp 算法已表明，由一定数量的已知位，可以很容易地预测后面的位。但是，仍然有许多质量较好的 pRNG 将 LFSR 的输出作为它们的原始输入。

伪随机数的另一个应用是**模拟**，出于这个目的，这个要求对重要的密码应用不算苛刻。

有些时候，一个确定的过程不能产生真正的随机数，实际的标准也不能衡量随机性。然而，可操作的标准就是：在不知道密钥的情况下，由随机数序列中已知的部分**推测**下一个数必须是“困难的”（对攻击者）。这里我们又将随机性问题转移到另一个问题上，即“推测”的含义又是什么呢？

稍微准确的讲，一个 pRNG 应当以一个**种子**或**密钥**作为输入，然后生成一个由种子完全决定的比特流或数的序列。

线性同余、线性反馈移位寄存器以及相关的发生器在密码学意义上都**不是**很好的，但它们仍可用于模拟。某些相关的发生器，称为*滞后的斐波那契发生器*，它们有相当长的周期，但都是不安全的。为了证明它们是弱的 pRNG，我们将给出对它们的有效攻击。

相比较而言，Blum-Blum-Shub 发生器被认为是相对安全的。因为它的安全性已被证明等价于模 n 二次剩余快速算法的存在性，这里的 n 为不知其分解的合数，而这个分解也被认为是困难的。但问题是用 BBS 算法产生每一个随机比特的代价相比其他算法要大一些。我们也会描述这个算法，以作比较。

21.1 假的一次一密乱码本

在我们称为异步流密码的应用中，通常的想法是一个大小适中的**密钥**生成一个看上去是随机且很长的模 m 的整数**密钥流**

$$S = (s_0, s_1, s_2, s_3, \cdots)$$

并把它们作为假的一次一密乱码本的密钥，这里的生成使用了一个选定的 pRNG（对某个模数 m）。

这也就是说要将密钥流 S 简单地与明文 $x=(x_0,x_1,x_2,\cdots)$ 相加（模 m）：

$$E_k(x)=(x_0+s_0,x_1+s_1,x_2+s_2,\cdots)$$

解密就是对应的减法。我们在这里之所以称其为假的一次一密乱码本，是因为这里的密钥流明显不满足使一次一密真正安全的“真随机性”要求。

如果由相对较短的密钥生成一个假的一次一密乱码本是可行的，那么这个系统会使一次一密乱码本的**密钥管理**简单化：不用为一次一密本分发数量巨大的密钥，只要将生成密钥流的密钥（更小一点）分发。比如下一份报文的密钥可以包含在每份报文的末尾。

在本文中，我们假设主要的密码问题是唯密文攻击。注意到如果密钥流是周期的，即它本身是重复的，那么这个密码就退化为维吉尼亚（Vigenere）密码。这是一个可根据字母频率很容易被攻击的密码。因此，对伪随机发生器的第一需要就是力图保证它的**周期尽可能的长**。

21.2 伪随机数发生器的周期

要求伪随机数发生器满足的第一个条件就是（对一给定的种子值）生成的序列不应当重复。那么这里就有一个序列**周期**的概念。这个概念的定义如下，一个序列 $s_0,s_1,s_2,s_3,\cdots$ 称为是**周期的**且**周期**为 p，如果对所有下标 i，有 $s_{i+p}=s_i$。

但是，这个定义还有一点限制。更一般地，称序列 $s_0,s_1,s_2,s_3,\cdots$ 是（**最终**）**周期**为 p 的周期序列，如果存在一个指标 i_0，使

$$s_{i+p}=s_i \quad (\text{对所有的 } i\geqslant i_0)$$

这样就允许在序列开始有一些不规则的动作，但最终这个动作是周期的。

当然，并不是每个序列都是周期的。但由一些简单的机制，如有限状态机，生成的序列倾向于是周期的。第一个问题就是使周期尽可能的大，这个问题的准确含义依赖于上下文具体的要求，在后面的各种例子中我们将具体说明。

21.3 同余发生器

线性同余发生器本身不适合用于秘密密钥密码，但对其他方面的应用却是有用的，比如模拟。尽管从密码意义上说它们不够好，但象其他许多失败的体制一样，理解它们是如何失败的都是有参考价值的。

最简单的**线性同余发生器**可由指定一个模数 m 和一个模 m 可逆的整数 a 得到。对一个 $\mathbf{Z}/m$ 中的**种子** s_0，伪随机数的序列按如下方式产生：

$$\begin{aligned}s_1&=a\cdot s_0\ \%\ m\\ s_2&=a\cdot s_1\ \%\ m\\ s_3&=a\cdot s_2\ \%\ m\\ &\vdots\\ s_{n+1}&=a\cdot s_n\ \%\ m\end{aligned}$$

更一般的，固定一个模数 m，一个模 m 的整数 a 及另一个整数 b，取**种子** $s_0\in\mathbf{Z}/m$，则伪随机数序列按如下方式产生

$$
\begin{aligned}
s_1 &= a \cdot s_0 + b \% m \\
s_2 &= a \cdot s_1 + b \% m \\
s_3 &= a \cdot s_2 + b \% m \\
&\vdots \\
s_{n+1} &= a \cdot s_n + b \% m
\end{aligned}
$$

但显然只要取 $b=0$ 就是前面那个较简单的发生器。

比如，取 $m=17, a=2, b=7$，则序列的下一个元素可由前面的一个元素按下面的方式生成

$$s_{n+1} = 2 \cdot s_n + 7 \% 17$$

取种子 $s_0=1$，则生成的伪随机数序列为

$$1, 9, 8, 6, 2, 11, 12, 14, 1, 9, 8, 6, 2, 11, 12, 14, 1, 9, \cdots$$

注意到如果选定任何一个起点，若干个数后它又一次出现，序列就开始重复。这是很显然的，因为下一个值就是完全由前面的值确定的。

在这个例子中，序列的**周期**是8，而且 $\mathbf{Z}/17$ 中的元素不是每个都在序列中出现，实际只出现了8个。

再来看一个例子，取 $m=17, a=7, b=1$，则序列的下一个元素由前面的一个元素以下面的方式生成：

$$s_{n+1} = 7 \cdot s_n + 1 \% 17$$

取种子 $s_0=1$，则生成的伪随机数序列为

$$1, 8, 6, 9, 13, 7, 16, 11, 10, 3, 5, 2, 15, 4, 12, 0, 1, 8, \cdots$$

跟前一个例子一样，一旦有一个值再一次出现，序列就开始重复。这是显然的，因为下一个值是完全由前面的值确定的。在这个例子中，模数同前，周期为16，模17的所有值中只有14没有出现。

如果用14做种子，我们会发现得到的序列为

$$14, 14, 14, 14, 14, 14, \cdots$$

这显然不是随机的。

无论数值 a, b, m, s_0 的哪一部分选取是保密的，它们都会被容易地从一小部分 $s_1, s_2, \cdots$ 中恢复出来。实际上，我们现在应该知道，有一个很大的密钥空间是必要的。但要保证一个密码的安全仅有这点是远远不够的，这种情况的例子就是维吉尼亚密码。

特别的，在维吉尼亚密码的讨论中，我们已指出了一个明显的重要问题：序列在重复之前应有多长？也就是说其周期有多长？即满足 $s_n = s_0$ 的最小整数 n 是多少？注意，如果 $s_n = s_0$，则 $s_{n+1} = s_1$，$s_{n+2} = s_2$，等等。如果随机序列的周期小于消息的长度，则这个密码就是一个维吉尼亚密码，它很容易受到 Friedman 重合指数攻击。关于周期长度的问题在做一些技术准备之后我们将给予详细回答。

除了有较长的周期外，还有一些准则应当满足。我们不准备解释这些准则，但仍要说明一个线性同余发生器在这个意义上是好的，因为它满足了许多尚未命名的准则。这个线性同余发生器出现在 *Comm. of ACM*, vol.31,1988, pp.1192～1201中。取

$$\text{模数 } m = 214783647$$

$$\text{乘子 } a = 16807$$

（显然这里的 b 为0）。给定一个种子 s_0，则生成的伪随机数序列为

$$s_0,(a\cdot s_0)\%\ m,(a^2\cdot s_0)\%\ m,(a^3\cdot s_0)\%\ m,\cdots$$

模数 m 为素数，而且 a 是模 m 的一个本原根。而且我们发现模数是一个梅森素数

$$2147483647=2^{31}-1$$

模 m 最小的本原根为 7，且 $a=16807=7^5$，而 5 不能整除 $m-1$，因此 7^5 仍为模 m 的一个本原根。最后，$a=7^5$ 相当接近于 m 的平方根，实际上要稍大一些。这给我们一种直觉的印象，乘以 a 后序列要“混合”的更好一些。

习题

21.3.01 考虑递归定义 $s_{n+1}=cs_n\%26$，这里 c 是模 26 的一个固定整数。证明由这样一个递归定义生成的密钥流的最大可能周期为 12。

21.3.02 考虑由下式定义的线性同余发生器 L

$$x_{n+1}=L(x_n)=7\cdot x_n+2\bmod 13$$

取初值 $x_0=6$。使 x_n 返回到初始值 $x_0=6$ 的最小整数 n 是多少？周期的长度是否依赖于初值？找出一个“坏的”初始值 x_{bad}，满足 $L(x_{\text{bad}})=x_{\text{bad}}$。

21.3.03 考虑由下式定义的线性同余发生器 L

$$x_{n+1}=L(x_n)=6\cdot x_n+9\bmod 11$$

取初值 $x_0=4$。使 x_n 返回到初始值 $x_0=4$ 的最小整数 n 是多少？周期的长度是否依赖于初值？找出一个“坏的”初始值 x_{bad}，满足 $L(x_{\text{bad}})=x_{\text{bad}}$。

21.3.04 考虑由下式定义的线性同余发生器 L

$$x_{n+1}=L(x_n)=7\cdot x_n+2\bmod 11$$

取初值 $x_0=6$。使 x_n 返回到初始值 $x_0=6$ 的最小整数 n 是多少？周期的长度是否依赖于初值？找出一个“坏的”初始值 x_{bad}，满足 $L(x_{\text{bad}})=x_{\text{bad}}$。

21.4 反馈移位发生器

现在我们来看一下另外一个随机数发生器，尽管这个发生器直接用于秘密密钥密码是不适合的，但对其他应用是有参考价值的。而且无论如何，我们必须知道它为什么不适合。固定一个数 N，一个模数 m（通常 $m=2$），选取**系数** $c=(c_0,c_1,\cdots,c_{N-1})$。另外选取一个**种子**或**初态** $s=(s_0,s_1,s_2,s_3,\cdots,s_{N-1})$，这个初态也是密钥流的开始。我们对 $n+1\geqslant N$，递归地定义：

$$s_{n+1}=c_0s_n+c_1s_{n-1}+c_2s_{n-2}+\cdots+c_{N-1}s_{n-N+1}\%m$$

这里用来定义密钥流的递归定义可以用矩阵来表示。为了简单起见，我们取 $N=4$，由系数 $c=(c_0,c_1,c_2,c_3)$，构造一个矩阵

$$C=\begin{pmatrix}c_0 & c_1 & c_2 & c_3\\ 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\end{pmatrix}$$

那么递归关系则可表示为

$$\begin{pmatrix} s_{n+1} \\ s_n \\ s_{n-1} \\ s_{n-2} \end{pmatrix} = C \cdot \begin{pmatrix} s_n \\ s_{n-1} \\ s_{n-2} \\ s_{n-3} \end{pmatrix} \quad (\text{均模 } m)$$

比如，假设模数 $m=2$，且系数 $c_0=1, c_1=0, c_2=0, c_3=1$，则输出序列由如下式子产生：

$$\begin{aligned} s_{i+1} &= c_0 \cdot s_i + c_1 \cdot s_{i-1} + c_2 \cdot s_{i-2} + c_3 \cdot s_{i-3} \\ &= 1 \cdot s_i + 0 \cdot s_{i-1} + 0 \cdot s_{i-2} + 1 \cdot s_{i-3} \\ &= s_i + s_{i-3} \end{aligned}$$

取种子 $(s_0, s_1, s_2, s_3) = (1,1,0,0)$，则生成的序列 (包括初态(1,1,0,0))为：

$$1,1,0,0,1,0,0,0,1,1,1,1,0,1,0,1,1,0,0,1,0,0,0,\cdots$$

在这个例子中，如果一个先前出现的4个连续比特又一次出现，则序列就开始重复了，因为前面四个值完全确定下一个比特。初态 (1,1,0,0) 在第15步后开始重复了。

用矩阵来表示输出序列流的计算，不仅对计算来说是方便的，而且便于作为一个密码攻击可利用的弱点。

选取不同的种子，可以得到不同周期的密钥流。有的选择可以得到长周期的密钥流，有的选择获得的则是周期较短的密钥流，当然系数 c 是相同的。出于这个考虑，在某些情况，系数 $c_0, c_1, c_2, \cdots$ 也应该小心谨慎地选择，而随机地选取种子 $(s_0, s_1, s_2, \cdots)$。为了深入研究周期长度的问题，还需做进一步的技术准备。

习题

21.4.01 考虑一个线性反馈移位寄存器，它由如下的递归表达式 $s_{n+1} = s_n - s_{n-1}$ 生成一个密钥流，s_i 为实数。证明不论选取什么样的种子 $s = (s_0, s_1)$，都将生成一个周期为6的周期密钥流（或者是6的一个因子）。

21.4.02 求一个常数 a_0，对某个初态 s_0, s_1，使如下递归式 $s_{n+1} = a_0 s_n + s_{n-1}\%26$ 生成一个周期长于6的密钥流。

21.4.03 求 LFSR 的周期，其初态为 $(x_0, x_1, x_2, x_3) = (0,1,0,0)$，定义为

$$x_{n+1} = x_n + x_{n-1} + x_{n-2} + x_{n-3}$$

21.4.04 L 是系数为 00101（升序）的线性反馈移位寄存器，并设初态为10000（按升序）。L 返回到它的初态时需要多少步？思考在初态再次出现前，不同的初态对步数的影响如何？

21.4.05 L 是系数为11101（升序）的线性反馈移位寄存器，并设初态为01000（按升序）。L 返回到它的初态时需要多少步？思考在初态再次出现前，不同的初始数据对步数的影响如何？你能猜出这与梅森数 $2^5 - 1$ 的初始状态有关吗？如果用移存器的系数作为一个五次多项式（首项为 x^5）的系数，那么这么五次多项式是否可约？是否一致？

21.5 Blum-Blum-Shub 发生器

与线性同余发生器、线性反馈移位寄存器和其他类型不安全的 pRNG 相比，BBS 发生器已被证明是安全的，前提条件是将一个大数分解为素数形式是困难的。由于这个关于安全性的证明相当长而且较难懂，这里我们就忽略掉它。

选取（秘密的）两个大素数 p 和 q，p 和 q 都满足模 4 同余 3，并计算 $n=p\cdot q$。模数 n 可以公开而且可以重复使用。给定一个**种子** s_0，用如下递归公式：

$$s_{i+1} = s_i^2 \% n$$

计算出序列 $s_1, s_2, s_3, \cdots$。用这个介于 0 至 $n-1$ 之间的序列，我们用下式得出一个伪随机比特序列：

$$b_i = s_i \%2$$

因为 p 是模 4 同余 3 的素数，对模 p 的一个平方 z，公式

$$y = z^{(p+1)/4} \bmod p$$

可以计算 z 的一个平方根，因为这个平方根是 z 模 p 的两个平方根之一，称其为**主平方根**。由此，平方映射是 p 平方集合的一个**置换**，对素数 q 亦有同样的结论，因此（由孙子定理），平方映射是模 n 平方集合的一个置换。

有一个详细的分析表明，如果由模 n 的 BBS 发生器生成的比特序列与一个随机比特序列有明显的区别，则存在一个快速的求解模 n 平方根的概率算法。实际上，这也就是给出了一个将 n 分解为 $n=p\cdot q$ 的快速概率算法。由于这个详细分析篇幅过长，这里就不复述了。

对于长度大于 512 比特（约 170 位的十进制数）的模数 n，在合理的时间内不存在分解 n 的算法，这也就是说 BBS 发生器是安全的。

（1999 年中期，Adi Shamir，即 RSA 中的那个“S”，设计了一个称为“Twinkle”的专用计算机，这个部分是模拟、部分是数字式的计算机可以比以往的计算机快 100 到 1000 倍的速度实施因子分解攻击。这使得 512 比特的 RSA 不再安全，因为任何*真正*关心并且具备一定资源的人都可在几个小时内分解 512 比特的 RSA 模数，而不是以往的十年。）

21.6 Naor-Reingold 发生器

Naor-Reingold 伪随机数发生器[Naor，Reingold 1995]还是比较新的。Naor-Reingold 发生器是 BBS 发生器的变形，而且与 BBS 发生器一样，如果假设分解整数 $N=p\cdot q$ 是不可行的，则 Naor-Reingold 发生器被证明是安全的。这里 p 和 q 均是比较大的素数，且均模 4 同余 3，该发生器定义如下：

固定 n，选取两个随机的 n – 比特素数 p 和 q，令 $N=p\cdot q$，然后再随机选取一个模 N 的平方 g，令

$$a = (a_{1,0}, a_{1,1}, a_{2,0}, a_{2,1}, \cdots, a_{n,0}, a_{n,1})$$

是一个长为 $2n$ 且值介于 1 到 N 的随机序列，$r=(r_1,\cdots,r_n)$ 是一个随机的 0，1 比特序列。对任何长度至少为 $2n$ 比特的整数 t，令 $\beta_{2n}(t)$ 表示 t 的二进制展开式的向量，必要时可以在左边填充 0 以使其长度达到 $2n$。比如 $2n=6, t=13$，t 表为二进制即 $13=1101_2$，因此

$$\beta_6(13) = (0,0,1,1,0,1)$$

左边的两个 0 是填充的。对这样两个二进制向量 $v=(v_1,\cdots,v_n)$，$w=(w_1,\cdots,w_n)$，其中 v_i 和 w_i 都为 0 或 1，由下式定义一种向量的模 2 点积：

$$v\cdot w = (v_1,\cdots,v_n)\cdot(w_1,\cdots,w_n) = (\sum_{1\leqslant i\leqslant n} v_i w_i)\%2$$

那么对 n 维 0,1 向量 $x=(x_1,\cdots,x_n)$，定义{0,1}值函数为

$$f(x)=f_{N,g,a,r}(x)=r\cdot\beta(g^{a_{1,x_1}+a_{2,x_2}+a_{3,x_3}+\cdots+a_{n,x_n}}\%N)$$

则我们有如下定理

定理（Naor-Reingold） 假设分解 N 是不可行的，则函数 $f(x)=f_{N,g,a,r}(r)$ 的输出与随机比特是没有区别的，这也就是说，序列

$$f(1),f(2),f(3),\cdots,f(t)$$

与一个随机比特序列没有区别，这里的 t 和 N 相比是很小的。

备注 定理的证明在某种程度上要比 BBS 发生器安全性的证明容易一些，但它的篇幅过长，这里就不证明了。

备注 在前面的讨论中我们用到了“随机”一词，其含义是给每个可能的选项赋予了一个概率，使得所有的选项有相同的概率，即这个概率分布是均匀的。并且必须假设无论选取什么样的机制，获得 Blum 整数 N 时，选取素数 p 和 q 在某种意义上说是充分随机的，或者使得分解 N 不可行。

备注 关于非随机比特与随机比特区别的问题值得我们进行深入的讨论。

21.7 线性同余发生器的周期

完全掌握一个线性同余发生器的周期只需要一点线性代数的知识。为了简便起见，我们只考虑模数为素数的情况。固定一个素数 p 为模数，取 $a\in\mathbf{Z}/p^{\times}$ 和 $b\in\mathbf{Z}/p$，首先考虑 $b=0$ 的情况，这比较容易。考虑由下式定义的线性同余发生器

$$s_{i+1}=a\cdot s_i\ \%p$$

定理 只要种子是模 p 非零的，则线性同余式发生器 $s_{i+1}=a\cdot s_i\ \%p$ 的**周期**为 a 在乘法群 $\mathbf{Z}/p^{\times}$ 中的阶。特别的，这个阶总是 $p-1$ 的一个因子，并且当 a 是一个模 p 本原根时，该阶为最大值 $p-1$。（我们已经知道这样的本原根是存在的，并且有 $\varphi(p-1)$ 个。）

证明 因为假设种子 s_0 是非零的（否则，序列中所有 s_i 都为 0），在序列中 $s_i=a^i\cdot s_0 \bmod p$ 没有一个是模 p 为 0 的。考虑周期为 ℓ 的情况：

$$s_{i+\ell}=s_i \quad （对所有充分大的 i）$$

由定义即得

$$a^{i+\ell}\cdot s_0=a^i\cdot s_0 \bmod p$$

两边除以 $s_0 \bmod p$，则得到

$$a^{i+\ell}=a^i \bmod p$$

消去 a^i 则变为 $a^{\ell}=1 \bmod p$。由拉格朗日定理（或根据费马小定理的基本计算）有 $\ell\mid p-1$。即 LCG 的周期等于 a 在 $\mathbf{Z}/p^{\times}$ 中的阶。我们已经证明了本原根的存在性，所以最大可能周期为 $p-1$。 ♣

下面我们再来考虑由 $s_{i+1}=a\cdot s_i+b\%p$ 所定义的更为一般的线性同余发生器，即在前面的基础上加了一个非零的常数 b，并假设 $a\neq 1 \bmod p$。

定理 线性同余发生器 $s_{i+1}=a\cdot s_i+b\%p$ 的**周期**为 a 在乘法群 $\mathbf{Z}/p^{\times}$ 中的阶，只要种子值 s_0 **不等于一个坏种子值**，$s_{\text{bad}}=-(a-1)^{-1}\cdot b \bmod p$。特别的，这个阶总是 $p-1$ 的因子，并且最大可能值为 $p-1$，即 a 为模 p 的本原根时达到最大值。（我们已知这样的本原根是存在的，且有 $\varphi(p-1)$ 个。）

证明　这个发生器可以用2×2矩阵表示如下：

$$\begin{pmatrix} s_{n+1} \\ s_n \end{pmatrix} = \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix} \begin{pmatrix} s_n \\ s_{n-1} \end{pmatrix}$$

反复使用这个式子，可以得到：

$$\begin{pmatrix} s_{n+k} \\ s_{n+k-1} \end{pmatrix} = \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix}^k \begin{pmatrix} s_n \\ s_{n-1} \end{pmatrix}$$

为了理解矩阵

$$L = \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix}$$

的作用，我们来求它的**特征值**和**特征向量**。即要找到一个数λ（特征值）和2×1矩阵v（特征向量）使得

$$Lv = \lambda \cdot v$$

也就是说我们要找到这样一个向量，乘法对它的作用尽可能地简单，尽管它就是一个标量乘法。

如果我们能准确地求得特征值，那么对应的特征向量可由解如下线性方程而获得：

$$Lv = \lambda \cdot v$$

凯莱-哈密尔顿定理已经告诉我们求一个$n \times n$矩阵M特征值的方法。设I是一个$n \times n$单位矩阵，令x为不定元，计算行列式：

$$P_M(x) = \det(xI - M)$$

这个多项式称为M的**特征多项式**。凯莱-哈密尔顿定理已表明，**特征方程**$P_M(x) = 0$的根就是M的特征值。

当然，在整个讨论中所有的计算都是模p的。

在我们讨论的线性同余发生器中，令I是2×2单位矩阵，计算

$$P_L(x) = \det\left(\begin{pmatrix} x & 0 \\ 0 & x \end{pmatrix} - \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix}\right)$$

$$= \det\begin{pmatrix} x-a & -b \\ 0 & x-1 \end{pmatrix} = (x-a)(x-1)$$

注意到特征多项式与b无关，因此特征方程就是$(x-a)(x-1)=0$，它的根为a和1。为求特征值a的特征向量$v = \begin{pmatrix} x \\ y \end{pmatrix}$，需要求解

$$\begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = a \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

该式可以化简为如下方程组

$$\begin{cases} a \cdot x + b \cdot y = a \cdot x \\ 0 \cdot x + y = a \cdot y \end{cases}$$

在第一个方程中消去$a \cdot x$，得$by = 0$。第二个方程可表为$(a-1)y = 0$，因为$a \neq 1$，所以$y = 0$。因此我们取$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$为$a$的特征向量。

接下来求特征值为1的特征向量 $v=\begin{pmatrix}x\\y\end{pmatrix}$，这需要求解

$$\begin{pmatrix}a & b\\0 & 1\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}=1\cdot\begin{pmatrix}x\\y\end{pmatrix}$$

该式可化简为如下方程组：

$$\begin{cases}a\cdot x+b\cdot y=x\\0\cdot x+y=y\end{cases}$$

第二个方程实际为 $0=0$，总是成立的。第一个方程可化简为 $(a-1)x+by=0$。

因为 $a-1\neq 0$，所以 $x=-(a-1)^{-1}by$，即 x 惟一地由 y 确定，我们取 $y=1$，即得到特征值1的特征向量 $\begin{pmatrix}-(a-1)^{-1}b\\1\end{pmatrix}$。

这里的用法是将每个向量表为特征向量标量乘积的和。这样在矩阵乘法下更容易理解它的作用。

比如我们可以将初态向量 $\begin{pmatrix}s_0\\1\end{pmatrix}$ 表为如下形式：

$$\begin{pmatrix}s_0\\1\end{pmatrix}=c\begin{pmatrix}1\\0\end{pmatrix}+d\begin{pmatrix}-(a-1)^{-1}b\\1\end{pmatrix}，\text{其中 } c,d\in\mathbf{Z}/p$$

由此式的第二个方程容易得到 $d=1$，由第一个方程可求解 c，即

$$s_0=c-(a-1)^{-1}b$$

则

$$c=s_0-(a-1)^{-1}b$$

现在我们来计算

$$L^k\begin{pmatrix}s_0\\1\end{pmatrix}=L^k\left((s_0+(a-1)^{-1}b)\begin{pmatrix}1\\0\end{pmatrix}+\begin{pmatrix}-(a-1)^{-1}b\\1\end{pmatrix}\right)$$

$$=a^k(s_0+(a-1)^{-1}b)\begin{pmatrix}1\\0\end{pmatrix}+\begin{pmatrix}-(a-1)^{-1}b\\1\end{pmatrix}$$

注意到上式的第二个被加项没有改变，因为它是对应于特征值1的。因此，如果上式的第一项为0，则生成的随机值将为常数。故坏种子值会出现，当且仅当

$$s_0=-(a-1)^{-1}b$$

这就证明了定理。 ♣

21.8 本原多项式

本原多项式的概念在许多应用中都是必不可少的。

定义 $\mathbf{Z}/p[x]$ 中的 N 次多项式 P 称为**本原**多项式，如果 $P\mid x^N-1$，但对 $0<t<N$ 的任何整数 t，P 不整除 x^t-1。另一个等价的定义为：N 次多项式 P 为本原多项式，当且仅当 $x^N=1 \bmod P$；但对 $0<t<N$，$x^t\neq 1 \bmod P$。

定理 N 次多项式 P 是本原的当且仅当

$$x^N=1 \bmod P$$

但对 $0<t<N$ 且 $t\mid N$，有 $x^t\neq 1 \bmod P$。

备注 定理表明，我们不必去全部验证小于 N 的指数，只需验证所有 N 的因子。

证明 只需证明 如果 $x^N = 1 \bmod P$，存在 $0 < t < N$ 使得 $x^t = 1 \bmod P$，那么对 N 的因子 d，$0 < d < N$，$x^d = 1 \bmod P$。设 a，b 为满足如下条件的整数

$$gcd(t,N) = aN + bt$$

则

$$x^{gcd(N,t)} = x^{aN+bt} = (x^N)^a \cdot (x^t)^b = 1^a \cdot 1^b = 1 \bmod P$$

注意对负的指数，我们需要知道 x 是模 P 可逆的。这一点的确是满足的，因为由 $x^N = 1 \bmod P$ 即可得 $x \cdot x^{N-1} = 1 \bmod P$。也就是说 x^{N-1} 是模 P 的逆元。所以总是有 $x^{gcd(N,t)} = 1 \bmod P$。命题得证。 ♣

稍后，我们还将证明如下定理。

定理 $\mathbf{F}_p[x]$中的 n 次多项式 P 是本原的，当且仅当 P 整除 $(p^n - 1)$ 阶分圆多项式。反之，$\mathbf{F}_p[x]$中 $(p^n - 1)$ 阶分圆多项式的每个不可约因子的次数为 n 且是本原多项式。

备注 注意到在该定理中声称 $(p^N - 1)$ 阶分圆多项式的不可约因子都是 N 次的，这一点并不明显。

我们还有如下对本原多项式计数的推论。

推论 $\mathbf{F}_p[x]$中 n 次本原多项式的数量为 $\varphi(p^n - 1)/n$，这里的 φ 是指欧拉函数。

证明 假设我们已证明了上面的定理，那么 $(p^n - 1)$ 阶分圆多项式的不可约因式恰为 n 次本原多项式。另一方面，我们已经知道 t 阶分圆多项式的次数为 $\varphi(t)$，因此 $(p^n - 1)$ 阶分圆多项式的次数为 $\varphi(p^n - 1)$。由定理的结论，这个分圆多项式为 n 次不可约多项式的乘积，因为多项式乘积的次数为被乘因式次数的和，则必然有 $(p^n - 1)$ 阶分圆多项式的 $\varphi(p^n - 1)/n$ 个不可约因子。由前面的讨论，这些因子恰为 n 次本原多项式。推论得证。 ♣

由上面的定理，本原多项式为不可约的。因此在对本原多项式计数时，只需对那些次数给定的不可约多项式计数。一个整数被称为是**无平方的**（square-free），如果它不能被任何素数的平方整除。回想一下墨比乌斯函数 μ 的定义：

$$\mu(n) = \begin{cases} (-1)^t & \text{如果}n\text{是无平方的，且恰被}t\text{个素数整除} \\ 1 & \text{若}n = 1 \\ 0 & \text{若}n\text{能被某个素数平方整除} \end{cases}$$

定理 $\mathbf{F}_p[x]$中首一的 n 次不可约多项式的数量为

$$\frac{1}{n} \sum_{1 \le d \le n, d|n} \mu(d) p^{n/d}$$

即这个数量为

$$\frac{1}{n} \cdot \left(p^n - \sum_{p_1|n} p^{n/p_1} + \sum_{p_1,p_2|n} p^{n/p_1p_2} - \sum_{p_1,p_2,p_3|n} p^{n/p_1p_2p_3} + \cdots \right)$$

这里的求和式包括了 n 的不同素因子的全体。（后面证明。）

该定理的一个特殊情况由下述推论给出发：

推论 如果 $n = p_1$ 为素数，则 $\mathbf{F}_p[x]$上首一的 n 次不可约多项式的个数为

$$\frac{p^{p_1}-p}{p_1}$$

该推论的证明很容易由上述定理得出。特别的，如果我们已接受了定理的结论，那么可以得到，次数为 n 且使得 (p^n-1) 为素数的所有不可约多项式为本原多项式。这必然要求 $p=2$ 或 $p-1$ 是一个真因子。 ♣

推论 设 n 为整数，且 2^n-1 为素数，则 $\mathbf{F}_2[x]$ 中每个 n 次不可约多项式都是本原多项式。

证明 前面的定理已经表明，每个 n 次本原多项式都是不可约的。在这里 n 次不可约多项式的个数就是首一不可约多项式的个数，由上述定理中的计数公式，有 $\frac{2^n-2}{n}$ 个 n 次不可约多项式。另一方面，由前面对首一本原多项式的计数公式，首一本原多项式的个数为 $\frac{\varphi(2^n-1)}{n}$。因为 2^n-1 是素数，则有

$$\varphi(2^n-1)=(2^n-1)-1=2^n-2$$

可见，两个计数方法得到了相同的结果。因为本原多项式集合是不可约多项式集合的一个子集，那么必有这两个集合相等。推论得证。 ♣

21.9 线性移位寄存器的周期

线性反馈移位寄存器（LFSR）完全可以用线性代数的方法来分析，这里需要用到**本原多项式**，设 $c_0,c_1,\cdots,c_{N-1}$ 为一个 LFSR 的系数，LFSR 按如下式子递归定义：

$$s_{n+1}=c_0s_n+c_1s_{n-1}+c_2s_{n-2}+\cdots+c_{N-1}s_{n-N+1}$$

定理 如果多项式 $x^N-c_0x^{N-1}-c_1x^{N-2}-c_2x^{N-3}-\cdots-c_{N-1}$ 为 $\mathbf{Z}/p[x]$ 上的本原多项式，则以 $c_0,c_1,\cdots,c_{N-1}$ 为系数的 LFSR 对任意的初始态 $s_0,s_1,\cdots,s_{N-1}$（非全为 0），其周期为 p^N-1。

备注 如果那个多项式不是本原的，则周期会较小，并且存在几个坏的初始状态，使得 LFSR 的周期比较小。

证明 同以前的例子一样，我们定义矩阵 L 如下：

$$L=\begin{pmatrix} c_0 & c_1 & c_2 & \cdots & c_{N-2} & c_{N-1} \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ & \cdots & & & & \\ 0 & 0 & 0 & & 1 & 0 \end{pmatrix}$$

即矩阵的最上面一行为 LFSR 的系数（注意它们的顺序），次对角线上元素为 1，其余皆为 0。这样我们有如下表示：

$$L\cdot\begin{pmatrix} s_n \\ s_{n-1} \\ \cdots \\ s_{n-N+2} \\ s_{n-N+1} \end{pmatrix}=\begin{pmatrix} s_{n+1} \\ s_n \\ \cdots \\ s_{n-N+1} \\ s_{n-N} \end{pmatrix}$$

在相对较简单的情况下，LFSR 的初态

$$\begin{pmatrix} s_{N-1} \\ s_{N-2} \\ \cdots \\ s_1 \\ s_0 \end{pmatrix}$$

可以表示为**线性组合**的形式，即

$$\begin{pmatrix} s_{N-1} \\ s_{N-2} \\ \cdots \\ s_1 \\ s_0 \end{pmatrix} = a_1 \cdot v_1 + \cdots + a_n \cdot v_n$$

这里的 v_i 是对应于特征值 λ_i 的**特征向量**，因此容易计算 L^k 为：

$$L^k \begin{pmatrix} s_{N-1} \\ s_{N-2} \\ \cdots \\ s_1 \\ s_0 \end{pmatrix} = \lambda_1^k a_1 \cdot v_1 + \cdots + \lambda_n^k a_n \cdot v_n$$

通过计算行列式，容易得到矩阵 L 的**特征多项式**正好是

$$P_L(x) = x^N - c_0 x^{N-1} - c_1 x^{N-2} - c_2 x^{N-3} - \cdots - c_{N-1}$$

但是我们不希望去解高次的多项式方程，所以也不能按 LCG 中的方法那样直接求解。这样，我们将要求所有的特征值 λ_i 有最大的可能的阶，也就是最小的正整数 ℓ，使得 λ_i^ℓ 尽可能的大。

备注 这里有一个问题，如果我们仅讨论有限域 $\mathbf{Z}/p$，而不是其他，则我们就无从去求解这些特征值，因为我们可能不能在 $\mathbf{Z}/p$ 上求解特征方程。尽管我们在 LFSR 的定义中没有提到有限域，但需要很好地理解在有限域上考虑 LFSR 会发生什么。但目前不必太担心这一点。

假设 P_L 是不可约的，那么这些特征值在有限域 F_{p^N} 中是存在的（因为 P_L 是 N 次的）。在本原根的讨论中，我们实际上已证明任何有限域的乘法群是循环的，所以 F_{p^N} 是阶为 $p^N - 1$ 的循环群。这也就是说 F_{p^N} 的每个非零元素满足

$$x^{p^N-1} - 1 = 0$$

但我们希望排除那些较小次的元素（由拉格朗日定理，有 P_L 真因子的次），也就是说我们希望考查从 $x^{p^N-1} - 1 = 0$ 中去掉所有 $x^d - 1$（d 为 $p^N - 1$ 的真因子）的公因子后剩下的部分。由分圆多项式的讨论，去掉这个公因式后剩下的恰好为 $(p^N - 1)$ 阶**分圆多项式**。

因此，假设 P_L 是不可约的且整除 $(p^N - 1)$ 阶分圆多项式能够保证每个特征值的阶为 $p^N - 1$。

下面来考查周期的条件

$$\lambda_1^{i+\ell} a_1 \cdot v_1 + \cdots + \lambda_n^{i+\ell} a_n \cdot v_n = \lambda_1^i a_1 \cdot v_1 + \cdots + \lambda_n^i a_n \cdot v_n$$

进一步化简得到

$$(\lambda_1^{i+\ell} - \lambda_1^i) v_1 + \cdots + (\lambda_N^{i+\ell} - \lambda_N^i) v_N = 0$$

或

$$(\lambda_1^{\ell}-1)\lambda_1^{i}v_1+\cdots+(\lambda_N^{\ell}-1)\lambda_N^{i}v_N=0$$

如果 p^N-1 能整除 ℓ，那么面和式中的每一项均为 0，则整个和为 0。

为了证明另外一半，即这个向量的和为 0，即指每个 $\lambda_i^{\ell}-1=0$，这里需要了解一点关于特征向量的知识。首先，在我们的证明中，P_L 有**不同的根**，这是因为 P_L 是 (p^N-1) 阶分圆多项式的因子，而我们知道分圆多项式有不同的根。现在我们将说明对于有不同特征值 $\lambda_1,\cdots,\lambda_N$ 的 $N\times N$ 矩阵 M，任何对应的特征向量 v_i 的组合关系

$$a_1\cdot v_1+\cdots+a_N\cdot v_N=0$$

则必有所有的系数 a_i 等于 0。为证明这个结论，我们假设有这样一种关系，它在所有这样的关系中有最少的非零 a_i。对那个等式两边同时作用 M，则得

$$\lambda_1 a_1\cdot v_1+\cdots+\lambda_N a_N\cdot v_N=0$$

将第一个关系式乘以 λ_j，并从第二个关系中减去这个关系，则得到如下向量关系

$$(\lambda_1-\lambda_j)a_1\cdot v_1+\cdots+(\lambda_N-\lambda_j)a_N\cdot v_N=0$$

这样做的结果是消去了第 j 项。注意，因为特征都是不同的，那么只有当 $i=j$ 时才会得到 $\lambda_i-\lambda_j=0$。因此就得到一个有更少非零系数的关系式，这和假设矛盾。

因此，在上面的讨论中，

$$(\lambda_1^{\ell}-1)\lambda_1^{i}\cdot v_1+\cdots+(\lambda_N^{\ell}-1)\lambda_N^{i}\cdot v_N=0$$

当且仅当所有系数 $(\lambda_i^{\ell}-1)\lambda_i=0$，因为 $\lambda_i^{p^N-1}=1$，λ_i 本身不为 0，所以这个充分必要条件就是对所有的 i，有 $\lambda_i^{\ell}-1=0$。因为每个 λ_i 的阶为 p^N-1，那么只有当且仅当 $\ell\mid p^N-1$。这也就是说我们已经证明了这样一个 LFSR 的阶为 p^N-1。 ♣

备注 我们没有证明每个向量都能表示为特征向量的如下线性组合

$$v=a_1\cdot v_1+\cdots+a_N\cdot v_N$$

因为特征值是互不相同的，这个结论是成立的，这里我们就不证明了。

21.10 本原多项式的例子

作为具体的例子，我们来确定 $\mathbf{F}_2[x]$ 中次数不超过 8 的本原多项式。我们使用的准则是 $\mathbf{F}_2[x]$ 中的 n 次本原多项式是 (2^n-1) 阶分圆多项式的不可约因子。这个准则对次数较小的多项式较为实用。稍后我们给出一种针对次数较大的多项式更加适用的计算准则。

$\mathbf{F}_2[x]$ 中的一个**线性**多项式是本原的，如果它不可约且整除 (2^1-1) 阶分圆多项式 $\varphi_1=x-1=x+1$。这已直接表明线性多项式 $x+1$ 是本原的，而线性多项式 x 却不是。

$\mathbf{F}_2[x]$ 中的一个**二次**多项式是本原的，如果它是不可约的且整除 (2^2-1) 阶分圆多项式

$$\varphi_{2^2-1}=\varphi_3=\frac{x^3-1}{x-1}=x^2+x+1$$

φ_3 本身就是二次的，容易验证 x^2+x+1 是不可约的，所以 x^2+x+1 就是模 2 惟一的二次本原多项式。

$\mathbf{F}_2[x]$ 中的一个**三次**多项式是本原的，如果它是不可约的且整除 (2^3-1) 阶分圆多项式

$$\varphi_{2^3-1}=\varphi_7=\frac{x^7-1}{x-1}=x^6+x^5+x^4+x^3+x^2+x+1$$

由试除法，我们已知模 2 的两个三次不可约多项式为 x^3+x^2+1 和 x^3+x+1。为验证本原性，

将两个多项式相乘：

$$(x^3+x^2+1)\cdot(x^3+x+1)=x^6+x^5+x^4+x^3+x^2+x+1$$

所以这两个三次不可约多项式都是本原的。

$\mathbf{F}_2[x]$中的一个**四次**多项式是本原的，如果它是不可约的且整除(2^4-1)阶分圆多项式

$$\varphi_{2^4-1}=\varphi_{15}=\frac{x^{15}-1}{(x^2+x+1)(x^5-1)}=x^8+x^7+x^5+x^4+x^3+x+1$$

由试除法，已知模2的四次不可约多项式恰有3个：

$$x^4+x^3+x^2+x+1,\quad x^4+x^3+1,\quad x^4+x+1$$

如果做个猜测，后面两个是本原的（因为它们看上去相互关联），那么我们验证

$$(x^4+x^3+1)\cdot(x^4+x+1)=x^8+x^7+x^5+x^4+x^3+x+1$$

因此，这两个是模 2 的本原多项式。因此，3 个不可约多项式中的两个是本原的，注意到$x^4+x^3+x^2+x+1$不是本原的，因为它实际上为φ_5。这意味着，它的每个根的阶为 5 而不是15。

$\mathbf{F}_2[x]$中一个**五次**多项式是本原的，如果它是不可约的且整除(2^5-1)阶分圆多项式

$$\varphi_{2^5-1}=\varphi_{31}=\frac{x^{31}-1}{x-1}$$

这是一个30次的多项式。因为它恰好是五次本原多项式的乘积，所以共有$30/5=6$个五次多项式。由试除法，我们可以找出这6个不可约的五次多项式：

$$x^5+0+x^3+x^2+x+1$$
$$x^5+x^4+0+x^2+x+1$$
$$x^5+x^4+x^3+0+x+1$$
$$x^5+x^4+x^3+x^2+0+1$$
$$x^5+x^3+1$$
$$x^5+x^2+1$$

因为存在6个不可约的五次多项式，并且φ_{31}也有6个不可约多项式，所以它们必定是相同的6个不可约多项式。如果表示怀疑，可以将这6个五次多项式相乘，看乘积是否为φ_{31}。

$\mathbf{F}_2[x]$中一个**六次**多项式是本原的，如果它是不可约且整除$(2^6-1)=63=3\cdot3\cdot7$阶的分圆多项式

$$\varphi(2^6-1)=\varphi(3\cdot3\cdot7)=(3-1)3(7-1)=42$$

因为φ_{63}恰为本原六次式的乘积，所以φ_{63}应当有$42/6=7$个六次本原多项式。但有多少个模2的六次不可约多项式呢，稍后我们将证明这个计数为

$$\frac{2^6-2^3-2^2+2^1}{6}=\frac{54}{6}=9$$

所以有两个不可约的六次式不是本原的，到底是哪两个？如果不进行大量的计算是难以下结论的。

$\mathbf{F}_2[x]$中一个**七次**多项式是本原的，如果它是不可约的且整除$(2^7-1)=127$阶的分圆多项式，其次数为

$$\varphi(2^7-1)=127-1=2\cdot3\cdot3\cdot7$$

我们可以猜测它是本原七次式的乘积，那么应该是$126/7=18$个七次式的乘积。但是$\mathbf{F}_2[x]$上的七次不可约多项式有多少个？事实表明有

$$\frac{2^7-2^1}{7}=\frac{126}{7}=18\text{个}$$

所以所有模2不可约的七次式都是本原的。

$\mathbf{F}_2[x]$中一个八次多项式是本原的，根据定义，如果它是不可约且整除$2^8-1=255$阶分圆多项式，φ_{255}的次数为$\varphi(2^8-1)=(3-1)(5-1)(7-1)=2\cdot4\cdot6=48$，如果我们期望它是本原的八次式的乘积，那么它应该是$48/8=6$个八次式的乘积。那么有多少个模2不可约的八次多项式呢？由我们后面要证明的公式，这个数量为

$$\frac{2^8-2^4}{8}=2^5-2=30$$

因此在30个不可约的八次式中仅有6个是本原的。

习题

21.10.01 在$\mathbf{F}_3$上找出两个首一且本原的二次多项式。

21.10.02 在$\mathbf{F}_5$上找出四个首一且本原的二次多项式。

21.10.03 在$\mathbf{F}_3$上找出四个首一且本原的三次多项式。

21.10.04(*) 找出$\mathbf{F}_2[x]$中的两个非本原的六次不可约多项式。

21.10.05(*) 找出$\mathbf{F}_2[x]$中的至少一个本原的八次多项式。

21.11 本原性检验

对于较大的次数d，我们需要有尽可能高效的方法检验$\mathbf{Z}/p[x]$上多项式的本原性。为达到这个目的，我们将进一步精练本原性的判别准则，还将给出几个例子。

定理 设P是$\mathbf{Z}/p[x]$中的n次多项式，令$N=p^n-1$，则P是本原的当且仅当

$$x^N=1 \bmod P$$

且对任何整除N的素数q，$x^{N/q}\neq 1 \bmod P$。

证明 我们已经知道，P是本原的，当且仅当$x^N=1 \bmod P$且对N的每个因子d，$0<d<N$，有$x^d\neq 1 \bmod P$。假设对N的某个因子d，$0<d<N$，有$x^d=1 \bmod P$。则$N/d>1$，故N/d有一个素因子，那么就有

$$x^{N/q}=x^{d\cdot(N/dq)}=(x^d)^{N/dq}=1^{N/dq}=1 \bmod P$$

注意因为$q \mid N/d$，所以N/dq是整数。定理得证。 ♣

备注 通常，检验一个多项式f是否**模P**等于1，就是用P去除f，看余式是否为1。

备注 当然，可以运用快速指数算法。

备注 $\mathbf{Z}/2[x]$中的快速指数算法运行起来特别快，因为它要做的就是对多项式平方以使指数翻倍。例如在$\mathbf{Z}/2[x]$中

$$(x^5+x^4+x+1)^2=x^{10}+x^8+x^2+1$$

例子 我们来检验九次多项式$1+x^4+x^9$为本原的，而$1+x+x^9$是不可约但非本原的。在两种情况下，我们都默认它们的不可约性。为了简便起见，我们将$\mathbf{Z}/2[x]$中的多项式表为非负整数的形式，整数i出现当且仅当x^i在多项式中出现（当然系数为$\mathbf{Z}/2$中元素）。

首先，注意到

$$2^9-1=7\cdot73$$

则（由拉格朗日定理）

$$x^7 = 1 \bmod \text{不可约的九次多项式}$$

或

$$x^{73} = 1 \bmod \text{不可约的九次多项式}$$

如果这个九次式是本原的，在第一种情况下，x^7 模任何九次式已经是最简化的，因此不可能有 $x^7 = 1 \bmod$ 不可约的九次多项式。在第二种情况下，用快速指数算法计算 $x^{73} \bmod 1 + x^4 + x^9$：

[1]	73	[0]
[1]	72	[1]
[2]	36	[1]
[4]	18	[1]
[8]	9	[1]
[8]	8	[0,4]
[2,6,7]	4	[0,4]
[0,3,5,7]	2	[0,4]
[1,4,6]	1	[0,4]
[1,4,6]	0	[4,6,8]

因此，

$$x^{73} = x^4 + x^6 + x^8 \bmod 1 + x^4 + x^9$$

下面我们来检验 $x^{511} = 1 \bmod 1 + x^4 + x^9$。当然，利用2的方幂，容易验证 $x^{512} = x \bmod 1 + x^4 + x^9$：

[1]	512	[0]
[2]	256	[0]
[4]	128	[0]
[8]	64	[0]
[2,6,7]	32	[0]
[0,3,5,7]	16	[0]
[1,4,6]	8	[0]
[2,3,7,8]	4	[0]
[0,2,5,7]	2	[0]
[1]	1	[0]
[1]	0	[1]

所以 $x^{512} = x \bmod 1 + x^4 + x^9$，故 $1 + x^4 + x^9$ 是本原的。

备注 如果对某个九次式 P，满足

$$x^{2^9} \neq x \bmod P$$

那么我们将知道这个九次式 P 是**可约的**（这是为什么？）。比如

$$\begin{aligned} P(x) &= 1 + x^9 = (1 + x^3)(1 + x^3 + x^6) \\ &= (1 + x)(1 + x + x^2)(1 + x^3 + x^6) \end{aligned}$$

我们来计算 $x^{512} \bmod P$，（这里有一个捷径）

$$x^{512} = x^{9\cdot 56+8} = (x^9)^{58}\cdot x^8$$
$$= 1\cdot x^8 = x^8 \bmod x^9+1$$

结果不为 $x \bmod x^9+1$，所以间接地证明了 x^9+1 不是不可约的。

最后，我们来验证前面的第二个九次式 $1+x+x^9$，用快速指数算法计算 $x^{73} \bmod 1+x+x^9$：

[1]	73	[0]
[1]	72	[1]
[2]	36	[1]
[4]	18	[1]
[8]	9	[1]
[8]	8	[0,1]
[7,8]	4	[0,1]
[5,6,7,8]	2	[0,1]
[1,2,3,4,5,6,7,8]	1	[0,1]
[1,2,3,4,5,6,7,8]	0	[0]

即

$$x^{73} = 1 \bmod 1+x+x^9$$

这就证明了 $1+x+x^9$ 不是本原的。

再计算一下 $x^{512} \bmod 1+x+x^9$：

[1]	512	[0]
[2]	256	[0]
[4]	128	[0]
[8]	64	[0]
[7,8]	32	[0]
[5,6,7,8]	16	[0]
[1,2,3,4,5,6,7,8]	8	[0]
[1,3,5,7]	4	[0]
[1,5]	2	[0]
[1]	1	[0]
[1]	0	[1]

因此 $x^{512} = x \bmod 1+x+x^9$，因为 $1+x+x^9$ 是不可约的。（为什么？）

习题

21.11.01 验证多项式 x^6+x^3+1 在 $\mathbf{F}_2[x]$ 中不是本原的。

21.11.02 验证多项式 $x^8+x^4+x^3+x+1$ 在 $\mathbf{F}_2[x]$ 中不是本原的。

21.11.03 验证多项式 $x^8+x^7+x^4+x^3+x^2+x+1$ 在 $\mathbf{F}_2[x]$ 中不是本原的。

21.11.04 证明：如果 2^d-1 是素数，则 $\mathbf{F}_2[x]$ 中任何 d 次不可约多项式必都是本原的。

为了更好地理解我们前面介绍过的 $\mathbf{Z}/p$（p 为素数）中平方根、立方根等公式，还需要进一步掌握和利用群的概念。在费马小定理的情形下，这些公式以猜想的形式出现。

但从整体上讲，它们被认为是必然的。

22.1 群同态

群 G 到另一个群 H 的一个函数（或映射）

$$f:G\to H$$

称为一个**群同态**，如果对所有的 g_1，$g_2\in G$，

$$f(g_1g_2)=f(g_1)f(g_2)$$

设 e_G 是 G 的单位元，e_H 是 H 的单位元，群同态 f 的**核**为

$$f\text{ 的核}=\ker f=\{g\in G:f(g)=e_H\}$$

f 的**像**跟一般函数的像类似：

$$f\text{ 的像 }\mathrm{im}f=\{h\in H:\text{存在}g\in G,\text{使得}f(g)=h\}$$

令 $f:G\to H$ 是一个群同态，e_G 是 G 的单位元，e_H 是 H 的单位元，则

- f 必将 G 的单位元映射到 H 的单位元，即 $f(e_G)=e_H$。
- 对 $g\in G$，$f(g^{-1})=f(g)^{-1}$。
- f 的核 $\ker f$ 是 G 的子群。
- f 的像 $\mathrm{im}f$ 是 H 的子群。
- 一个群同态 $f:G\to H$ 是单射，当且仅当 f 的核是平凡的（即 $\ker f$ 为平凡子群 $\{e_G\}$）。

证明　G 的单位元 e_G 在 f 下的像 $f(e_G)$ 有如下性质：

$$f(e_G)=f(e_G\cdot e_G)=f(e_G)\cdot f(e_G)$$

利用群 G 单位元的性质以及群同态的性质，左乘 $f(e_G)^{-1}$，则得到

$$f(e_G)^{-1}\cdot f(e_G)=f(e_G)^{-1}\cdot(f(e_G)\cdot f(e_G))$$

化简整理后得到：

$$e_H=(f(e_G)^{-1}\cdot f(e_G))\cdot f(e_G)=e_H\cdot f(e_G)=f(e_G)$$

这就证明了 f 将 G 的单位元映射到 H 的单位元。

为了验证逆元的像就是像的逆元，由群同态的性质，只要简单地计算

$$f(g^{-1})\cdot f(g)=f(g^{-1}\cdot g)$$

再由逆元的性质以及单位元映射为单位元的事实得

$$f(g^{-1}\cdot g)=f(e_G)=e_H$$

同理可计算

$$f(g)\cdot f(g^{-1})=e_H$$

即逆元的像就是像的逆元得证。

为了证明群同态 $f:G\to H$ 的核 $\ker f$ 是 G 的子群，我们必须证明三件事。首先，必须验证 $\ker f$ 中单位元的存在性，这可由 $f(e_G)=e_H$ 直接得到。其次，必须证明如果 $g\in\ker f$，则 $g^{-1}\in\ker f$。用上面已经证明的结论，$f(g^{-1})=f(g)^{-1}$，如果 $f(g)=e_H$，则

$$f(g^{-1})=f(g)^{-1}=e_H^{-1}=e_H$$

最后，假设 $x,y\in\ker f$，则

$$f(xy)=f(x)\cdot f(y)=e_H\cdot e_H=e_H$$

所以核中元素的“乘积”还在核中。

现在设 X 是 G 的子群，令

$$f(X)=\{f(x):x\in X\}$$

为了证明 $f(X)$ 是 H 的子群，我们也必须验证三件事：单位元的存在性，对取逆元的封闭性以及对乘法的封闭性。首先，我们已经证明 $f(e_G)=e_H$，子群的像包含单位元。其次，$f(g^{-1})=f(g)^{-1}$，即子群的像对逆是封闭的。最后，由 $f(xy)=f(x)\cdot f(y)$，根据群同态的定义，像对乘法亦是封闭的。

最后，我们来证明同态 $f:G\to H$ 是单射当且仅当 $\ker f$ 是平凡的。首先，如果 f 是单射，则至多有一个元素可被映射到 $e_H\in H$。因为我们已经知道通过该同态至少有 e_G 被映射到 e_H，因此，必然只有 e_G 被映射到 e_H。所以 $\ker f=\{e_G\}$ 是平凡的。

另一方面，假设 $\ker f$ 是平凡的。我们将由假设 $f(x)=f(y)$，证明 $x=y$。对 $f(x)=f(y)$ 左乘 $f(g)^{-1}$，则得到

$$e_H=f(x)^{-1}\cdot f(x)=f(x)^{-1}\cdot f(y)$$

由群同态的定义，

$$e_H=f(x)^{-1}\cdot f(y)=f(x^{-1}y)$$

因此 $x^{-1}y\in\ker f$，由假设 $x^{-1}y=e_G$，左乘 x，即得 $y=x$。这就证明了单射性。 ♣

如果群同态 $f:G\to H$ 是**满射**，则称 H 是 G 的**同态像**。如果群同态 $f:G\to H$ 是**双射**，则称 f 为一个**同构**，G 和 H 称为**同核的**。

备注 从理论上看，同核的两个群可以被看成是“相同的”，从这个意义讲，任何本质群论的论断如果在一个群上是正确的，则在另一个同核群上亦成立。但是，在实践中，经由同核的结构转变是难以计算的。

习题

22.1.01 由 $\mathbf{Z}$（加法）到 $\mathbf{Z}/N$（模 N 加法）的同态 $x\to x \bmod N$ 的核是什么？

22.1.02 设 M,N 为正整数，$N\mid M$，那么由 $\mathbf{Z}/M$（模 M 加法）到 $\mathbf{Z}/N$（模 N 加法）的映射 $x \bmod M\to x \bmod N$ 的核是什么？

22.1.03 令 $\det:GL(2,\mathbf{Q})\to\mathbf{Q}^\times$ 为普通的行列式

$$\det\begin{pmatrix}a & b\\ c & d\end{pmatrix}=ad-bc$$

通过计算证明 det 是一个群同态。

22.1.04 对任意整数 n 和正整数 N，由 $f(x)=n\cdot x$ 定义的映射 $f:\mathbf{Z}/N\to\mathbf{Z}/N$ 是一个群同态（模 N 加法）。

22.1.05 对任意整数 n 和正整数 N，由 $f(x)=x^n$ 定义的映射 $f:\mathbf{Z}/N^\times\to\mathbf{Z}/N^\times$ 是一个群

同态。

22.1.06 固定一个正整数 N，证明对任意群同态 $f:\mathbf{Z}/N\to\mathbf{Z}/N$（模 N 加法）存在一个整数 n，使得 $f(x)=n\cdot x$。(提示：$n=f(1)$，并利用 $f(x)=f(\underbrace{1+\cdots+1}_{x})$)。

22.1.07 证明映射 $t\to\begin{pmatrix}1 & t\\ 0 & 1\end{pmatrix}$ 是 $\mathbf{Q}$ 到 $GL(2,\mathbf{Q})$ 的一个子群的同构。

22.1.08 证明映射 $\begin{pmatrix}a & b\\ 0 & d\end{pmatrix}\to a$ 是所有矩阵 $\begin{pmatrix}a & b\\ 0 & d\end{pmatrix}$ 的群到乘法群 $\mathbf{Q}^{\times}$ 的同态。这里 a，d 为非零的有理数，b 为任意的有理数，$\mathbf{Q}^{\times}$ 为非零有理数集合，该同态的核是什么？

22.1.09 证明 $\begin{pmatrix}a & b\\ 0 & d\end{pmatrix}\to b$ 不是一个同态。

22.1.10 定义映射 $E:\mathbf{Q}\to GL(2,\mathbf{Q})$ 为 $x\to\begin{pmatrix}1 & x\\ 0 & 1\end{pmatrix}$。证明 E 是由 $\mathbf{Q}$（加法）到 $GL(2,\mathbf{Q})$ 的子群的群同态。

22.1.11 定义映射 $E:\mathbf{Q}\to GL(3,\mathbf{Q})$ 为 $x\to\begin{pmatrix}1 & x & \frac{x^2}{2}\\ 0 & 1 & x\\ 0 & 0 & 1\end{pmatrix}$。证明 E 是由 $\mathbf{Q}$（加法）到 $GL(3,\mathbf{Q})$ 的子群的群同态。

22.1.12 定义映射 $r:\mathbf{R}\to GL(2,\mathbf{R})$ 为 $x\to\begin{pmatrix}\cos x & \sin x\\ -\sin x & \cos x\end{pmatrix}$。证明 r 是由 $\mathbf{R}$（加法）到 $GL(2,\mathbf{R})$ 的子群的群同态。

22.1.13 设 n 是整数，证明定义为 $f(x)=nx$ 的映射 $f:\mathbf{Z}\to\mathbf{Z}$ 是同态。

22.1.14 证明一个同态 $f:G\to H$ 总是具有性质：$f(g^{-1})=f(g)^{-1}$，$g\in G$。

22.2 有限循环群

一个有限群 G 是**循环群**，如果存在 $g\in G$，使 $\langle g\rangle=G$。这个元素 g 称为 G 的**生成元**，G 称为是**由 g 生成的**（无限循环群放在下一节讨论）。有限循环群是所有群中最简单的，并且容易被理解的一类群。

令 $N=|G|$,因为 $G=\langle g\rangle$，则 $N=|g|$。如下一些结论（以前已经证明过的）是很重要的。

- 元素 $e=g^0,g^1,g^2,\cdots,g^{N-2},g^{N-1}$ 构成了 $G=\langle g\rangle$ 的不同元素列表。
- 对任意整数 i,j，$g^i=g^j$ 当且仅当 $i\equiv j \bmod N$。
- 给定整数 j，设 i 是 j 模 N 后的约简，则 $g^j=g^i$。

有限循环群的所有子群和所有生成元均可完全掌握：

- G 的不同子群恰好就是子群 $\langle g^d\rangle$，d 为 N 的所有因子。
- 对于 $d\mid N$，子群 $\langle g^d\rangle$ 的阶就是 g^d 的阶 N/d。
- 对于任意整数 $k\neq 0$，g^k 的阶为 $N/gcd(k,N)$。
- 对任意整数 n，我们有 $\langle g^n\rangle=\langle g^{gcd(n,N)}\rangle$。

- G 的不同生成元为元素 g^r，这里 $1\leqslant r<N$ 且 $gcd(r,N)=1$。因此共有 $\varphi(N)$ 个生成元，φ 指欧拉函数。
- 在阶为 N 的有限循环群中，阶为 n 的元素个数为零，除非当 $n|N$，则个数为 N/n。

备注 这些性质可以用文字来表述。例如，一个有限循环群的每个子群仍是有限循环群，且其阶整除群的阶。反之，对群的阶的每个因子，存在这个阶的惟一子群。

证明 我们来证明 g^k 的阶为 $N/gcd(k,N)$。首先，如果 $(g^k)^\ell=e=g^0$，则 $k\ell\equiv 0 \bmod N$，即 $N|k\ell$。也就是存在一个整数 m，使 $k\ell=mN$，用 $gcd(k,N)$ 同时除等式两边，得

$$\frac{k}{gcd(k,N)}\cdot\ell=m\cdot\frac{N}{gcd(k,N)}$$

因为 $N/gcd(k,N)$ 和 $k/gcd(k,N)$ 是互素的，由惟一分解定理 $\frac{N}{gcd(k,N)}|\ell$，因此 g^k 实际的阶为 $N/gcd(k,N)$ 的倍数。另一方面

$$(g^k)^{N/gcd(k,N)}=(g^N)^{k/gcd(k,N)}=e^{k/gcd(k,N)}=e$$

注意，这里用到了 $N/gcd(k,N)$ 和 $k/gcd(k,N)$ 均为整数这个事实，所以这里的表示均是有意义的，这也就证明了 g^k 的阶是 $N/gcd(k,N)$。

作为一种特殊情况，如果 $k|N$，则 g^k 的阶为 $N/gcd(k,N)=N/k$。

我们已经知道，对任意 h 有 $|\langle h\rangle|=|h|$，因此当然就有

$$\left|\left\langle g^k\right\rangle\right|=\left|g^k\right|=N/gcd(k,N)$$

给定整数 k，我们来证明

$$\left\langle g^k\right\rangle=\left\langle g^{gcd(k,N)}\right\rangle$$

设 $d=gcd(k,N)$，并设 s,t 是满足 $d=sk+tN$ 的整数，则

$$g^d=g^{sk+tN}=(g^k)^s\cdot(g^N)^t=(g^k)^s\cdot(e)^t=(g^k)^s\cdot e=(g^k)^s$$

所以 $g^d\in\left\langle g^k\right\rangle$，另一方面，$g^k=(g^d)^{k/d}$。因为 $d|k$，所以 $g^k\in\left\langle g^d\right\rangle$，因此子群 $\left\langle g^k\right\rangle$ 和 $\left\langle g^d\right\rangle$ 对乘法和逆都是封闭的。对任意整数 ℓ

$$(g^k)^\ell\in\left\langle g^d\right\rangle$$

且

$$(g^d)^\ell\in\left\langle g^k\right\rangle$$

但 $\left\langle g^d\right\rangle$ 刚好是 g^d 的所有整数方幂的集合，对 g^k 也一样。因此，有

$$\left\langle g^d\right\rangle\subset\left\langle g^k\right\rangle$$

反之亦然，所以有

$$\left\langle g^d\right\rangle=\left\langle g^k\right\rangle$$

因此，$G=\langle g\rangle$ 的所有循环子群就是那些形如 $\left\langle g^d\right\rangle$ 的子群，其中 $d|N=|G|=|g|$。不同的因子 d 给出不同的子群。

设 H 是 G 的任意子群，我们必须证明 H 是由某个 g^k 生成的。令 k 为满足 $g^k\in H$ 的最小正整数，我们称 $\left\langle g^k\right\rangle=H$。对任意的 $g^m\in H$，可以将 m 表为如下形式：

$$m = q \cdot k + r \qquad 0 \leqslant r < k$$

因为 H 是一个子群，所以

$$g^r = g^{m-q\cdot k} = g^m /(g^k)^q \in H$$

又因为 k 是满足 $g^k \in H$ 的最小正整数，且 $0 \leqslant r < k$，则必有 $r = 0$。因此 m 是 k 的倍数，g^k 生成 H。

另一特殊情形，注意到 $\langle g^k \rangle = \langle g \rangle$ 当且仅当 $gcd(k,N)=1$。我们只需要考虑 $0<k<N$，否则就出现重复了。$\langle g \rangle$ 的不同生成元是那些满足 $0<k<N$，且 $gcd(k,N)=1$ 的元素 g^k，这样的元素当然有 $\varphi(N)$ 个。

同样，因为

$$\left|g^k\right| = \left|\left\langle g^k \right\rangle\right| = \left|\left\langle g^{gcd(k,N)} \right\rangle\right| = \left|g^{gcd(k,N)}\right|$$

所以就不难计算 $\langle g \rangle$ 中给定阶元素的个数了。 ♣

- 有限循环群的同态像仍是有限循环群。

证明　这只需要验证生成元的像是像的生成元即可。 ♣

- 阶为 N 有限循环群同构于 $\mathbf{Z}/N$。特别的，对有限循环群 G 的任意生成元 g，映射

$$f: n \to g^n$$

定义了同构映射 $f: \mathbf{Z}/N \to G$。

证明　这正好是上述某些性质的解释。 ♣

这里有一个可能造成干扰的问题，就是要证明上述定义的映射 f 是**定义明确的**。也就是说，有某种公式表面上看似描述了一个映射，但都存在着隐藏缺陷。我们必须证明如果 $m = n \bmod N$，则 $f(m) = f(n)$。（这与内射性无关）这个结论似乎是显然的，因为我们已经（在循环子群的讨论中）证明 $g^m = g^n$ 当且仅当 $m = n \bmod N$。

强调一点，我们将循环群 G 中的群运算记为 $*$ 而不是加法或乘法。这样关键在于证明同态性质 $f(m+n) = f(m) * f(n)$。实际上

$$f(m+n) = f((m+n)\%N) = g^{m+n\%N} = g^{m+n}$$

因为已经证明 $g^i = g^j$ 当且仅当 $i = j \bmod N$。上式也就等于 $f(g^m) * f(g^n)$。

为证明 f 是单射，假设 $f(m) = f(n)$，m,n 为整数，则 $g^m = g^n$。这也就意味着 $m = n \bmod N$，即 $m - \text{mod} - N = n - \text{mod} - N$。所以 f 是单射。

满射容易证明，给定 $g^n \in \langle g \rangle$，$f(n) = g^n$。因此 f 是双射且同态，所以 f 是同构。

习题

22.2.01 给出 $\mathbf{Z}/8$ 中阶为 4 的所有元素，以及 $\mathbf{Z}/72$ 所有阶为 6 的元素。

22.2.02 给出 $\mathbf{Z}/13$ 中阶为 4 的所有元素。

22.2.03 证明 $\mathbf{Z}/100$ 中由 3，97 生成的子群 $\langle 3 \rangle$ 和 $\langle 97 \rangle$ 是同一个子群。

22.2.04 假设 $G = \langle g \rangle$ 是一个阶为 30 的循环群，计算元素 $g^4, g^8, g^{12}, g^{16}, g^{20}, g^{24}, g^{28}$ 的阶。

22.2.05 假设 G 是一个有限循环群，且共有 2 个子群，它本身和平凡子群 $\{e\}$。那么 G 的阶如何确定？

22.2.06 假设 G 是一个有限循环群，且共有 3 个子群，它本身、平凡子群 $\{e\}$ 和一个阶为 13 的真子群，那么 G 的阶是多少？

22.2.07 p 为素数，假设群 G 有 p 个元素，证明 G 是循环群。（提示：取 $g \neq e$ 并考虑 $\langle g \rangle$，利用拉格朗日定理）。

22.2.08 设 m,n 互素，设 H,K 是群 G 的子群，这里 $|H|=m$，$|K|=n$。证明 $H \cap K = \{e\}$。

22.3 无限循环群

非有限的循环群是存在的，尽管它不同于有限循环群，但它的本质仍是简单的。循环群是有限的这一假设产生了一些复杂性，但仍是可以处理的。比如，加法群 $\mathbf{Z}$ 是一个无限循环群。如果 G 是无限群且存在一个 $g \in G$，使 $\langle g \rangle = G$，群 G 则称为**无限循环群**。这样的元素 g 是 G 的一个**生成元**，G 称为是**由 g 生成的**。

类似于前面的有限循环群，掌握下述关于无限循环群的结论是很重要的。

- 元素 $\cdots, g^{-3}, g^{-2}, g^{-1}, g^0 = e, g^1 = g, g^2, g^3, \cdots$ 为群 $\langle g \rangle = G$ 的所有不同元素。
- 对于整数 i,j ，$g^i = g^j$ 当且仅当 $i=j$。

一个无限循环群同构于 $\mathbf{Z}$。特别的，对无限循环群的任何生成元 g，映射 $g^n \to n$ 是 G 到 $\mathbf{Z}$ 的一个同构映射。因此，由于 $\mathbf{Z}$ 有两个生成元，所以无限循环群刚好有两个生成元。

下述结论有助于我们认识无限循环群的所有生成元及所有子群。

- G 的不同子群恰为子群 $\langle g^d \rangle$，d 为非负整数。
- 任何子群 $\langle g^d \rangle$ 是无限循环群，除了平凡子群 $\{e\} = \{g^0\} = \langle g^0 \rangle$。
- 每个子群 $\langle g^d \rangle$ 恰有两个生成元，g^d 和 g^{-d}。

这些结论的某些方面可以用文字表述：无限循环群的非平凡子群仍是一个无限循环群。

关于不同阶的元素的个数，我们有这样的结论：一个无限循环群的所有元素都是无限阶的，除了 $e = g^0$，它的阶为 1。

习题

22.3.01 对 $\mathbf{Z}$ 的任何非平凡子群 $\{0\}$ 的 H，证明 H 中最小的正整数为 H 的生成元，且 H 是一个循环群。

22.4 群中的根和方幂

在一个阶为 n 的循环群 $G = \langle g \rangle$ 中，关于方程 $x^r = y$ 的可解性已经有了明确的结论。

设 G 是一个以 g 为生成元的 n 阶循环群，给定整数 r，定义函数 $f: G \to G$，$f(x) = x^r$。

定理 映射 f 是 G 到自身的一个群同态。如果 $gcd(r,n)=1$，则 f 是一个同构。也就是说，如果 $gcd(r,n)=1$，则每个 $g \in G$，均存在一个 r 次根并且仅有一个这样的根。通常，

$$|\ker f| = gcd(r,n)$$

$$|\mathrm{im} f| = n / gcd(r,n)$$

如果一个元素 y 有一个 r 次根，则它确有 $gcd(r,n)$ 个 r 次根，且在 G 中有 $n/gcd(r,n)$ 个 r 次方幂。

证明 由

$$f(x \cdot y) = (xy)^r = x^r y^r = f(x) \cdot f(y)$$

即知 f 是一个同态（其中推导过程用到了 G 为阿贝尔群的性质）。

我们可以用 G 与 $\mathbf{Z}/n$ 同构的结论，这样就可以用有关 $\mathbf{Z}/n$ 的性质以及模 n 加法的简单作用来证明任意有限循环群的性质。因此，将 $\mathbf{Z}/n$ 加法的加法记号作一转换，映射 f 就是

$$f(x)=r\cdot x$$

我们已经知道如果 $gcd(r,n)=1$，则对 r 模 n 存在一个乘法逆元 r^{-1}。因此函数 $g(x)=r^{-1}\cdot x$ 就是函数 f 的逆。这就证明了 f 既是单射，又是满射，故它是一个同构。

对任意的 r，给定 y，我们来考查方程

$$r\cdot x=y \bmod n$$

的可解性。也就是说

$$n\mid(rx-y)$$

或者对某个整数 m，

$$mn=rx-y$$

令 $d=gcd(r,n)$，这样必定有 $d\mid y$，否则方程无解。另一方面，假设 $d\mid y$，对某个整数 y'，$y=dy'$，我们需要求解

$$r\cdot x=dy' \bmod n$$

除以 d 公因子，这个同余式变为

$$\frac{r}{d}\cdot x=y' \bmod \frac{n}{d}$$

消去公因子后，r/d 和 n/d 是互素的。所以存在 $r/d \bmod n/d$ 的乘法逆元 $(r/d)^{-1}$，且

$$x=(r/d)^{-1}\cdot y' \bmod (n/d)$$

也就是说满足这个条件的任意整数 x 是 $rx=y \bmod n$ 的一个解。设 x_0 是这样一个解，则整数

$$x_0, x_0+\frac{n}{d}, x_0+2\cdot\frac{n}{d}, x_0+3\cdot\frac{n}{d}, \cdots, x_0+(d-1)\cdot\frac{n}{d}$$

是解且模 n 是不相同的。这样总共有 d 个不同的模 n 解。

方程 $rx=y \bmod n$ 有一个解的充分必要条件 $gcd(r,n)\mid y$ 也表明，恰好存在 $n/gcd(r,n)$ 个模 n 的整数 y 满足这个条件。这也就是说恰好有 $n/gcd(r,n)$ 个“r 次幂”。

f 的核是那些满足 $rx=0 \bmod n$ 的 x 的集合，去掉公因子 $d=gcd(r,n)$，即得 $(r/d)x=0 \bmod n/d$，即 $(n/d)\mid(r/d)x$。因为 r/d 和 n/d 没有公因子，则由惟一分解定理，$n/d\mid x$。因此，模 n 有 d 个不同的解 x，即 $\ker f$ 有 d 个不同的元素。定理证毕。 ♠

在前面的章节中我们已经讨论过，若 p 是模 4 同余 3 的素数，则容易得到(模 p 平方的)平方根：有一个公式

$$b\text{的平方根}=b^{(p+1)/2} \bmod p$$

显然，这个公式给出了模 p 的一个平方根，仅当 b 是模 p 的平方。其他情况则得到一些无用的结果。下面这个定理就是这个结构的抽象。

定理 设 G 是一个阶为 N 的阿贝尔群，n 是一个与 N 互素的整数。令 r 是 n 模 N 的乘法逆元，对任何 $x\in G$，G 中恰有一个 x 的 n 次根，且它可由下式给出：

$$\sqrt[n]{x}=x^r$$

备注 通常，在没有前提保证这样一个结论存在之前，使用记号 $\sqrt[n]{x}$ 是不可取的。

证明 因为 r 是 n 模 N 的乘法逆元，所以我们对某个整数 ℓ 有

$$r\cdot n=1+\ell N$$

则

$$(x^r)^n = x^{rn} = x^{1+\ell N} = x\cdot(x^N)^\ell$$

由拉格朗日定理的推论，任何元素 x 的阶是群的阶 N 的一个因子，故 $x^N = e$。则进一步得到：

$$(x^r)^n = x^{rn} = x^{1+\ell N} = x\cdot(x^N)^\ell = x\cdot e^\ell = x\cdot e = x$$

这就证明了 x 的 n 次根是存在的，且由指定的公式给出。

为了证明惟一性，我们假设 y 和 z 都是 x 的 n 次根，则有

$$(y/z)^n = y^n/z^n = x/x = e$$

在这里，我们用到了群的交换性（阿贝尔性质）：如果不具备交换性，则通常 $(xy)^k = x^k y^k$ 是不成立的。故 y/z 的阶是 n 的一个因子。由拉格朗日定理，这个阶也应当是 N 的一个因子，故 y/z 的阶是 n 和 N 的公因子。但 $gcd(n,N)=1$，所以这个阶只能是 1，即 $y/z=e$，由此 $y=z$。至此定理得证。 ♣

现在我们可以用公式来表示循环群上 n 次根的**抽象欧拉准则**。这也是对古典数论中具体情况的抽象。

定理 设 G 是一个阶为 N 的循环群，n 为整数且 $n|N$，令 $m=N/n$，一个给定元素 $g\in G$ 在 G 中有一个 n 次方幂（即存在一个 $h\in G$，使 $h^n=g$）当且仅当

$$g^m = e$$

证明 设 $G=\langle x\rangle$，如果 g 是元素 $h=x^\ell$ 的 n 次方幂，则 $g=(x^\ell)^n$，且

$$g^m = ((x^\ell)^n)^m = (x^\ell)^{mn} = (x^\ell)^N = e$$

这是因为（由拉格朗日定理）任何元素的 N 次方幂都为 e。

另一方面，假设 $g^m=e$，则对某个指数 ℓ，$g=x^\ell$，

$$e = g^m = (x^\ell)^m = x^{\ell m}$$

因为 x 的阶是 N，所以 $e=x^{\ell m}$ 成立当且仅当 $N|\ell m$。又因为 $m=N/n$，所以得到 $n|\ell$（分解惟一），即

$$g = x^\ell = x^{n(\ell/n)} = (x^{\ell/n})^n$$

这表明 g 是一个 n 次方幂。定理得证。 ♣

习题

22.4.01 设 p 是一个模 4 同余 3 的素数，并设 a 是 $\mathbf{Z}/p$ 中的一个平方，证明 $a^{(p+1)/4}$ 是 a 的一个平方根。

22.4.02 设 p 是一个模 9 同余 7 的素数，如果 a 是 $\mathbf{Z}/p$ 中的一个立方，证明 $a^{(p+2)/9}$ 是 a 的一个立方根。

22.4.03 设 p 是一个模 9 同余 4 的素数，并设 a 是 $\mathbf{Z}/p$ 中的一个立方，证明 $a^{(p+5)/9}$ 是 a 的一个立方根。

22.5 平方根算法

设 G 是一个阶为 N 的循环群，我们考虑 $2|N$ 的情形，并考查一个求平方根的一般算法。

首先，我们必须找出 G 中的非平方元素 g，这是该算法的概率部分。随机选取一个元素 $g_0\in G$，计算 $g_0^{N/2}$（用快速指数算法）。如果 $g_0^{N/2}\neq e$，则取 $g=g_0$；如果 $g_0^{N/2}=e$，则 g_0 就是一个平方。再随机选取一个不同的 $g_1\in G$，重复上述计算步骤，直到找到一个非平方元

素 g 为止。

备注 我们正在利用偶数阶循环群上平方的抽象欧拉准则。

备注 注意每次随机选取的 $g_i \in G$，均有 50% 的机会可能是一个非平方元素。由前面的讨论，我们知道由 x 生成的偶数阶 N 循环群 $G=\langle x\rangle$ 的不重复元素为：

$$x^0, x^1, x^2, \cdots, x^{N-1}$$

并且这些元素中，只有那些指数为偶数，即 x^{ℓ}，$2|\ell$ 的元素才是平方。因此，共有 $N/2$ 个元素 $x^0, x^2, \cdots, x^{N-4}, x^{N-2}$ 是平方，而 $N/2$ 个元素 $x^1, x^3, \cdots, x^{N-3}, x^{N-1}$ 为非平方。

设 2^s 是能够整除 N 的 2 的最大方幂，那么就有

$$N = 2^s \cdot m$$

其中 m 为奇数。设 H 是 G 的一个包含 2^s 次方幂的子群。由循环群上抽象的欧拉定理，$h \in G$ 在 H 中当且仅当 $h^m = 1$。也就是说 H 的阶为奇数 m，因此每个 $h \in H$ 就有惟一的平方根，由下式给出

$$\sqrt{h} = h^{(m+1)/2}$$

给定 G 中的一个平方 y 以及一个非平方 g，如果对某个**偶数** e 和 $h \in H$，将 y 表示为

$$y = g^e \cdot h$$

则

$$\sqrt{y} = g^{e/2} \cdot h^{(m+1)/2}$$

我们需要的就是求指数 e 的一个简便算法。（有时我们得到的 e 是二进制形式的，如同在快速指数算法中的那样）。

p 为素数，则计算模 p 平方 b 的平方根的步骤如下：

- 将 $p-1$ 分解为 $p-1=2^s \cdot m$ 的形式，m 为奇数；
- 找到一个模 p 的非平方元素 g；
- 令 $B = b^m \bmod p$，$G = g^m \bmod p$，初始值 $E=0$；
- 对 $i=2,\cdots,s$，如果 $(B \cdot G^E)^{2^{s-i}} \neq 1 \bmod p$，就用 $E + 2^{i-1}$ 代替 E，否则 E 保持不变；
- 令 $F = (p-1) - E$；
- 则 $\sqrt{b} = (b \cdot g^E)^{(m+1)/2} \cdot g^{F/2} \bmod p$。

例 计算 $\sqrt{11} \bmod 1013$。令 $b=11, p=1013$，首先找出一个模 1013 的非平方元素 g，一般用猜测加欧拉准则验证的方法：猜测 2 为非平方元素，计算

$$2^{(1013-1)/2} = -1 \bmod 1013$$

因此 g=2 是一个模1013的非平方数。从1013－1中提取出 2 的方幂：

$$1013 - 1 = 2^2 \cdot 235$$

所以令 $s=2, m=253$，计算

$$B = b^m = 11^{253} \% 1013 = 1012 = -1 \bmod 1013$$

$$G = g^m = 2^{253} \% 1013 = 45 \bmod 1013$$

令 $E=0$，指数 i 的范围是 $i=2,\cdots,s$，这里只有一个指数 $i=2$：

$$(B \cdot G^E)^{2^{s-i}} = (-1)^{2^{2-2}} = (-1)^{2^0} = (-1)^1 \neq 1 \bmod 1013$$

所以用 $E + 2^{i-1}$ 代替 E，

$$E + 2^{i-1} = E + 2^{2-1} = E + 2^1 = E + 2 = 0 + 2 = 2$$

此时 E=2，然后代入 $F = (p-1) - E = 1012 - 2 = 1010$

$$\begin{aligned}\sqrt{11} &= (b\cdot g^{E})^{(m+1)/2}\cdot g^{F/2}\\ &= (11\cdot 2^{2})^{(253+1)/2}\cdot 2^{1010/2}\\ &= 44^{127}\cdot 2^{505}\\ &= 32 \bmod 1013\end{aligned}$$

最后，我们可以验证

$$32^{2} = 1024 = 1024 - 1013 = 11 \bmod 1013$$

所以就找到了一个11模1013的平方根。

例 计算 $\sqrt{2} \bmod 48049$。令 $b=2, p=48049$，首先找出一个模48049的非平方元素 g，一般用猜测加欧拉准则验证的方法：这里2,3,5,7,11,13均是模48049的平方，17是最小的非平方数：

$$17^{(48049-1)/2} = -1 \bmod 48049$$

所以 $g=17$ 是模48049的非平方，从 $48049-1$ 中提取2的方幂：

$$48049 - 1 = 2^{4}\cdot 3003$$

故令 $s=4, m=3003$，计算

$$B = b^{m} = 2^{3003}\%48049 = 23300 \bmod 48049$$

$$G = g^{m} = 17^{3003}\%48049 = 12891 \bmod 48049$$

令 $E=0$，指数 i 的范围是 $i=2,\cdots,s$，这里则为 $i=2,3,4$

$$B^{2^{s-2}} = (23300)^{2^{4-2}} = 48048 \neq 1 \bmod 48049$$

所以用 $E+2^{i-1}$ 代替 E，

$$E + 2^{i-1} = E + 2^{i-1} = E + 2^{1} = E + 2 = 0 + 2 = 2$$

对 $i=3$：

$$(B\cdot G^{E})^{2^{s-3}} = (23300\cdot 12891^{2})^{2^{4-3}} = 1 \bmod 48049$$

（故在这一步 E 保持不变），最后一步 $i=4$：

$$\begin{aligned}(B\cdot G^{E})^{2^{s-4}} &= (23300\cdot 12891^{2})^{2^{4-4}}\\ &= (23300\cdot 12891^{2})^{1} = 1 \bmod 48049\end{aligned}$$

（所以在这一步 E 保持不变），最后的 E 仍为 $E=2$。令

$$F = (p-1) - E = 48049 - 1 - 2 = 48046$$

$$\begin{aligned}\sqrt{2} &= (b\cdot g^{E})^{(m+1)/2}\cdot g^{F/2}\\ &= (2\cdot 17^{2})^{(3003+1)/2}\cdot 17^{48046/2}\\ &= 578^{1502}\cdot 17^{24023}\\ &= 310 \bmod 48049\end{aligned}$$

我们可以验证

$$310^{2} = 96100 = 2 \bmod 48049$$

因此310的确是2模48049的一个平方根。

习题

22.5.01 利用概率算法求 2 模 17 的一个平方根。
22.5.02 利用概率算法求 211 模 433 的一个平方根。
22.5.03 利用概率算法求 331 模 449 的一个平方根。
22.5.04 利用概率算法求 123 模 521 的一个平方根。
22.5.05 利用概率算法求 67 模 569 的一个平方根。

第23章 伪素性证明

终于我们可以来证明概率素性检验是不是真正能够起作用。首先我们证明卡米克尔数的一些相对直观的性质，这些还不难理解。另外一些证明则比较难，在不影响后续内容理解的情况下，我们略去了证明过程。另一方面，这些证明充分运用了抽象代数的理论。

23.1 λ 函数

理解卡米克尔数即意味着要理解费马伪素数检验是如何失败的。为掌握这种机制，我们继续利用欧拉定理和费马小定理的内容。

对于一个正整数 n，定义卡米克尔数的 λ **函数**如下：

$$\lambda(n) = \text{乘法群 } \mathbf{Z}/n^{\times} \text{ 的指数}$$

这个用法和任意群 G 的指数的概念是完全一致的，即 $\lambda(n)$ 是满足对每个 $x \in \mathbf{Z}/n^{\times}$，使得 $x^{\lambda(n)} = 1 \bmod n$ 的最小正整数。

在循环子群的讨论中我们已经知道，如果 $x^k = 1 \bmod n$，则元素 x 的阶 $|x|$ 整除 k：为了回想一下这个重要结论的证明过程，我们将 k 表示为

$$k = q \cdot |x| + r, 0 \leqslant r < |x|$$

则有

$$1 = x^k = x^{q \cdot |x| + r} = (x^{|x|})^q \cdot x^r = (1)^q \cdot x^r = x^r \bmod n$$

因为 $|x|$ 是满足 $x^{|x|} = 1 \bmod n$ 的最小正整数，则必有 $r = 0$，所以 $|x|$ 整除 n。

由拉格朗日定理我们还知道，一个有限群的指数整除群的阶，因此

$$\lambda(n) = \text{乘法群 } \mathbf{Z}/n^{\times} \text{ 的指数} \mid \varphi(n)$$

现在我们就可以完全决定 $\lambda(n)$ 了，尽管它不是弱乘性的，但它的作用还是能够被控制的。

定理

- 若 m 与 n 互素，则卡米克尔的 λ 函数满足 $\lambda(m \cdot n) = lcm(\lambda(m), \lambda(n))$。
- 若 p 为奇素数，则 $\lambda(p^e) = \varphi(p^e) = (p-1)p^{e-1}$。因为存在一个模 p^e 的本原根，所以存在一个元素，它的阶为 $\lambda(p^e)$。
- 对于 2 的方幂：$\lambda(2) = 1$，$\lambda(4) = 2$，$\lambda(2^e) = 2^{e-2}$，这里 $e > 2$。$\mathbf{Z}/2^{e\times}$ 中存在一个阶为 $\lambda(2^e)$ 的元素。

证明　若 m 与 n 互素，则由孙子定理，同余方程组

$$\begin{cases} x^k = 1 \bmod m \\ x^k = 1 \bmod n \end{cases}$$

就等价于单个同余式 $x^k = 1 \bmod mn$。因此，如果有 $x^k = 1 \bmod mn$，则当然有 $x^k = 1 \bmod m$ 和 $x^k = 1 \bmod n$。故由陈述定理前的讨论，$\lambda(m)$ 和 $\lambda(n)$ 都整除 $\lambda(mn)$，所以 $\lambda(mn)$ 是 $\lambda(m)$ 和 $\lambda(n)$ 的公倍数。另一方面，设 M 是 $\lambda(m)$ 和 $\lambda(n)$ 的任意公倍数，记 $M = m'\lambda(m)$，$M = n'\lambda(n)$，m' 和 n' 为整数，则我们有

$$x^M = (x^{\lambda(m)})^{m'} = 1^{m'} = 1 \bmod m$$
$$x^M = (x^{\lambda(n)})^{n'} = 1^{n'} = 1 \bmod n$$

因此由孙子定理（m 与 n 互素），$x^M = 1 \bmod mn$。这也就是说，如果 M 能被模 m 的所有阶和模 n 的所有阶整除，那么 M 就能被模 mn 的所有阶整除。这就证明了

$$\lambda(m \cdot n) = lcm(\lambda(m), \lambda(n))$$

对于 p 为奇素数的情况，我们已知存在一个模 p^e 的本原根 g。因此在 $\mathbf{Z}/p^{e\times}$ 中存在一个阶为 $(p-1)p^{e-1} = \varphi(p^e)$ 的元素。故 $\varphi(\ ^e)$ 整除 $\lambda(p^e)$。由拉格朗日定理，$\mathbf{Z}/p^{e\times}$ 中的任何元素的阶都整除 $\mathbf{Z}/p^{e\times}$ 的阶 $\varphi(p^e)$。再一次利用陈述定理前的讨论，即可得 $\lambda(p^e) = \varphi(p^e)$。

现在来考虑 2 的方幂的情况。群 $\mathbf{Z}/2^\times$ 只有一个元素，所以它的阶必然是 1。$\mathbf{Z}/4^\times$ 的阶为 2，自然有 2 阶的元素。

下面我们来证明对 $e \geqslant 3$，不存在模 2^e 的本原根。我们注意到对任意整数 x，

$$(1+2x)^2 = 1 + 4x(1+x) = 1 \bmod 8$$

更进一步，我们有

$$(1+8x)^{2^k} = 1 \bmod 2^{3+k}$$

因此，对于 $e \geqslant 3$ 有

$$(1+2x)^{2^{e-2}} = 1 \bmod 2^e$$

由此，任何元素实际的阶必为 2^{e-2} 的因子。另一方面，对于 $k \geqslant 2$ 以及奇整数 x，

$$(1+2^k x)^2 = 1 + 2^{k+1}x + 2^{2k}x^2 = 1 + 2^{k+1}\underbrace{(x + 2^{k-1}x^2)}_{y} = 1 + 2^{k+1}y$$

因为 $k \geqslant 2$，$2^{k-1}x^2$ 为偶数，所以 $y = x \bmod 2$。因此由归纳法，

$$(1+4x)^{2^l} = 1 + 2^{2+l}y$$

并且 $y = x \bmod 2$。特别地，前面的模数 2^e 不为 1，除非 $2+l \geqslant e$。比如，元素 $1+4$ 在 $\mathbf{Z}/2^{e\times}$ 中的阶为 2^{e-2}。因此，对 $e \geqslant 3$，$\lambda(2^e) = 2^{e-2}$。这就完成了 $\lambda(n)$ 的计算。 ♣

23.2 卡米克尔数

尽管存在无限多个卡米克尔数（Carmichael numbers），使人们难以应付，但也存在一些明显的限制，表明了什么的数才是卡米克尔数。这些限制在证明索洛维-斯特拉森和米勒-罗宾检验为什么能够成功检验合数性方面发挥了很明显的作用。

定理 一个正整数是卡米克尔数，当且仅当 $\lambda(n) \mid n-1$。特别地，一个卡米克尔数必定是奇数、无平方且至少被三个不同的素数整除。

证明 假设 n 是卡米克尔数，即假设对每个与 n 互素的整数 b，有

$$b^{n-1} = 1 \bmod n$$

由定义，对每个与 n 互素的整数 b，卡米克尔函数 $\lambda(n)$ 是满足 $b^{\lambda(n)} = 1 \bmod n$ 的最小正整数。我们已经知道存在一个阶为 $\lambda(n)$ 的元素，设 b 就是这个元素，记 $n-1 = q \cdot \lambda(n) = r$，这里 $0 \leqslant e < \lambda(n)$。由此则有

$$1 = b^{n-1} = b^{q\cdot\lambda(n)+r} = (b^{\lambda(n)})^q \cdot b^r \bmod n$$

因为 $\lambda(n)$ 是满足 $b^{\lambda(n)} = 1 \bmod n$ 的最小正整数，则必有 $r = 0$，所以 $\lambda(n) \mid n-1$。

从 $\lambda(n)$ 的一般公式，我们注意到，如果 $n > 2$，则 $2 \mid \lambda(n)$。如果 n 有任何不同于 2 的素因子 p，则 $p-1 \mid \lambda(n)$。另一方面，如果 n 是大于 2 的方幂，仍有 $2 \mid \lambda(n)$。

因此，如果n是卡米克尔数，那么因为$2|\lambda(n)$且$\lambda(n)|n-1$，则$2|n-1$，所以n为奇数。

如果对某个奇素数p，$p^e|n$，这里$e>1$，则$p|\lambda(n)$。因为$\lambda(n)|n-1$，则$p|n-1$。但当$p|n$时，这不可能成立。因此n是无平方的。

如果n正好是两个不同的奇素数p和q的乘积，即$n=pq$，则有

$$\lambda(n)=\lambda(p\cdot q)=lcm(\lambda(p),\lambda(q))=lcm(p-1,q-1)$$

而$\lambda(n)|n-1=pq-1$，所以得到$p-1|pq-1$和$q-1|pq-1$。我们将$p-1|pq-1$稍做一点变形，即有

$$p-1|pq-1=(p-1)q+q-1$$

因此，$p-1|q-1$，同理可得$q-1|p-1$。但是因为p和q为不同的素数，故这不可能。因此n必然至少被三个不同的素数整除。 ♣

23.3 欧拉证据

对照卡米克尔数没有合数性的费马证据的事实，我们现在来证明非素数的合数性的欧拉证据是存在的。我们还会证明对合数存在许多欧拉证据，在$1<b<n$之间至少有一半的数b是n的合数性的欧拉证据。

命题 n的一个素性的欧拉证据也是n的素性的一个费马证据。而n的合数性的一个费马证据也是一个合数性的欧拉证据。换句话说，一个错误的素性欧拉证据是一个错误的素性费马证据。特别地，如果存在一个合数n，它没有其合数性的欧拉证据，则n必为一个卡米克尔数。

证明 我们在命题中的断言也就是说，如果对与n互素的整数b，有

$$b^{(n-1)/2}=\left(\frac{b}{n}\right)_2 \bmod n$$

那么则有

$$b^{n-1}=1 \bmod n$$

实际上对第一个等式两边平方，即得

$$b^{n-1}=\left(\frac{b}{n}\right)_2^2 \bmod n$$

因为b与n互素，二次符号（勒让德符号）的值为±1，则其平方必为 1。因此欧拉证据b当然是一个费马证据。

如果n是一个合数，且使对所有与n互素的整数b，有

$$b^{(n-1)/2}=\left(\frac{b}{n}\right)_2 \bmod n$$

对此数b仍有$b^{n-1}=1 \bmod n$，且n是卡米克尔数。 ♣

备注 因此我们可以说{欧拉伪素数}⊂{费马伪素数}，或者更准确一点，{欧拉伪素数基b}⊂{费马伪素数基b}。

现在我们来证明合数性的欧拉证据的存在性。

定理（欧拉证据的存在性） 假设n是一个正的合数，则至少存在一个整数b，$1<b<n$，且$gcd(b,n)=1$，使得

$$b^{(n-1)/2} \neq \left(\frac{b}{n}\right)_2$$

即存在一个欧拉证据。

证明 如果 n 不是卡米克尔数，则已经存在一个 n 的合数性的费马证据 b，那么 b 当然也是欧拉证据。

所以考虑一个卡米克尔数 n，由前面的结论我们知道，n 是奇数且是无平方的，所以记作 $n = pm$，p 为素数且 p 不整除 m。令 b_0 是模 p 的一个非二次剩余，且（由孙子定理）求出 b，使 $b \equiv b_0 \bmod p$ 且 $b \equiv 1 \bmod m$。则一方面，利用雅可比符号的定义

$$\left(\frac{b}{n}\right)_2 = \left(\frac{b}{pm}\right)_2 = \left(\frac{b}{p}\right)_2\left(\frac{b}{m}\right)_2 = \left(\frac{b_0}{p}\right)_2\left(\frac{1}{m}\right)_2 = (-1)(+1) = -1$$

可得

$$\left(\frac{b}{pm}\right)_2 = \left(\frac{b}{p}\right)_2\left(\frac{b}{m}\right)_2$$

另一方面，

$$b^{(n-1)/2} = 1^{(n-1)/2} = 1 \bmod m$$

仍使用模 m 的运算，还有

$$b^{(n-1)/2} \neq \left(\frac{b}{pm}\right)_2 \bmod m$$

因此当然有

$$b^{(n-1)/2} \neq \left(\frac{b}{pm}\right)_2 \bmod pm$$

因此，这个 b 是 $n = pm$ 合数性的一个欧拉证据。 ♣

利用拉格朗日定理，我们可以证明存在“许多”欧拉证据。

推论 对于合数 n，在 $1 < b < n$ 中至少有一半的数 b 是 n 的合数性的欧拉证据。

证明 我们的主要思想就是证明错误证据（但与 n 互素）的集合 L 是 $Z/n^{\times}$ 的一个子群。因为由定理得知至少存在一个证据，因此错误证据的子群是一个真子群。由拉格朗日定理，L 的阶 $|L|$ 必定是 $Z/n^{\times}$ 的阶 $\varphi(n)$ 的一个真因子。因此

$$|L| \leqslant \frac{1}{2}\varphi(n) \leqslant \frac{1}{2}(n-1)$$

现在我们来证明 n 的素性证据的集合 L 是 $Z/n^{\times}$ 的一个子群。假设 x, y 是 n 的素性证据，即

$$x^{(n-1)/2} = \left(\frac{x}{n}\right)_2 \bmod n$$

$$y^{(n-1)/2} = \left(\frac{y}{n}\right)_2 \bmod n$$

那么

$$(xy)^{(n-1)/2} = x^{(n-1)/2} \cdot y^{(n-1)/2} = \left(\frac{x}{n}\right)_2 \cdot \left(\frac{x}{n}\right)_2 \bmod n$$

而我们已经知道$\left(\frac{x}{n}\right)_2 \cdot \left(\frac{y}{n}\right)_2 = \left(\frac{xy}{n}\right)_2$。

因此，

$$(xy)^{(n-1)/2} = x^{(n-1)/2} \cdot y^{(n-1)/2} = \left(\frac{x}{n}\right)_2 \cdot \left(\frac{y}{n}\right)_2 = \left(\frac{xy}{n}\right)_2 \bmod n$$

这也就是说 xy 也是 n 的素性的一个证据。

下面，我们来验证 $\mathbf{Z}/n^{\times}$ 中乘法单位元1属于 L，这很容易得到：

$$1^{(n-1)/2} = 1 = \left(\frac{1}{n}\right)_2 \bmod n$$

再下来我们验证证据的集合 L 对取模 n 的乘法逆是封闭的。设 $x \in L$，x^{-1} 表示模 n 的乘法逆元。首先，我们有

$$\left(\frac{x}{n}\right)_2 \cdot \left(\frac{x^{-1}}{n}\right)_2 = \left(\frac{x \cdot x^{-1}}{n}\right)_2 = \left(\frac{1}{n}\right)_2 = 1$$

因此

$$\left(\frac{x^{-1}}{n}\right)_2 \cdot \left(\frac{x}{n}\right)_2 = 1$$

然后再利用指数的性质：

$$(x^{-1})^{(n-1)/2} = (x^{(n-1)/2})^{-1} = \left(\frac{x}{n}\right)_2^{-1} = \left(\frac{x^{-1}}{n}\right)_2 \bmod n$$

也就是说，如果 x 是一个证据，则 x^{-1} 也是一个证据。

因此，L 对乘法和逆封闭且包含单位元，所以是 $\mathbf{Z}/n^{\times}$ 的一个子群。因为我们已经证明对合数 n，至少存在一个 b，它不是 n 的素性的一个证据。那么对于合数 n，错误证据的子群是一个*真子群*。因此，如同证明开始时所指出的那样，由拉格朗日定理，我们断定至少有一半的整数 b，$1 < b < n$，将检测合数 n 的合数性。 ♣

由此，我们确信索洛维–斯特拉森素性检验能够发挥作用。

23.4 强证据

现在我们来证明强证据的存在性，以及实际上对奇合数 n，至少有 3/4 的整数 b 是 n 的合数性的证据，其中 $1 < b < n$。因此，米勒-罗宾检验可以发挥作用。用这种方法，我们可以拿强伪素数和欧拉伪素数作比较。

在本节的讨论中，n 为一固定的奇整数，且令 $n-1 = 2^s \cdot m$，m 为奇数。因为 n 为奇数，故 $s \geqslant 1$。

我们来回顾一下用单个辅助整数 b 的米勒-罗宾检验方法。首先，选取一个“随机”辅助数 b，$1 < b < n$，并计算 $c = b^m$。如果 $c = 1 \bmod n$，则停止：b 就是 n 的素性的一个**强证据**。如果 $c \neq -1 \bmod n$，则计算连续的平方：

$$c^2, c^4 = (c^2)^2, c^8 = ((c^2)^2)^2, c^{2^4} = (c^8)^2, \cdots, c^{2^{s-1}}$$

如果对任何 $k < s$，我们得到 $c^{2^k} = -1 \bmod n$，则停止：b 就是 n 的素性的一个**强证据**。另一方

面，如果对某个 k，我们得到 $c^{2^k}=1 \bmod n$ 但 $c^{2^{k-1}} \neq -1 \bmod n$，则 n 为**确切合数**。如果对 $0 \leqslant k \leqslant s$，不存在 $c^{2^k}=1$，则 n 也为**确切合数**。

备注 如果对 $0 \leqslant k \leqslant s$，不存在 $c^{2^k}=1$，则 $b^{n-1} \neq 1 \bmod n$，故 n 对费马伪素数基 b 检验失败。

如果使用 k 个辅助数 $b_1, b_2, \cdots, b_k$，并且每一个都是 n 的素性的证据，则我们设想 n 为素数的“概率”为 $1-\left(\frac{1}{4}\right)^k$。

我们应当验证一个真正的素数对任何基 b 是一个强伪素数。即我们可以验证，一个真正的素数将通过米勒–罗宾检验。

命题 真素数是强伪素数，且通过米勒-罗宾检验。即设 p 是大于 2 的素数，且 $p-1=2^s \cdot m$，m 为奇数。令 b 是不被 p 整除的任何整数，t 是满足 $(b^m)^{2^t}=1 \bmod p$ 的最小非负整数，则 $t=0$，或者 $(b^m)^{2^{t-1}}=-1 \bmod p$。

证明 因为 p 是素数，$\mathbf{Z}/p$ 是一个域，且由代数基本定理，方程 $x^2-1=0$ 根的个数最多等于它的次数。因此，$\pm 1-\text{mod}-p$ 是 $\mathbf{Z}/p$ 中惟一平方为 $1-\text{mod}-p$ 的元素。所以，如果 $(b^m)^{2^t} \neq 1 \bmod p$，则惟一的可能就是 $(b^m)^{2^t}=-1 \bmod p$。因此，真素数 p 将通过每一个这种检验。 ♣

我们有一个简单的结论将表明，米勒-罗宾检验至少与费马检验是有区别的。这个事实跟强证据是欧拉证据一起，也是如下结论的推论：欧拉证据是费马证据。但这个命题本身有一个更简单的证明。

命题 n 的素性的一个强证据也是一个费马证据。

证明 设 b 是一个随机选取的数，$c=b^m$。首先假设 c=1,则

$$b^{n-1}=b^{m \cdot 2^s}=c^{2^s}=1^{2^s}=1 \bmod n$$

所以 b 是一个费马证据。否则，如果 $c^{2^t}=-1 \bmod n$，$t<s$，则

$$b^{n-1}=b^{m \cdot 2^s}=c^{2^s}=(c^{2^t})^{2^{s-t}}=(-1)^{2^{s-t}}=1 \bmod n$$

b 仍是一个费马证据。 ♣

在证明强证据是欧拉证据之前，我们需要一些记号以及预先的计算，对正整数 N 和 k，

$$\text{ord}_N k = \text{乘法群} \mathbf{Z}/N^{\times} \text{中} k \text{的阶}$$

引理 k 为整数，p 为奇素数，且 p 不整除 k，则 $\text{ord}_{p^e} k / \text{ord}_p = p$ 的非负方幂。

证明 我们利用 $\mathbf{Z}/p^{ex}$ 为循环群这一事实，即存在一个本原根 g。设 l 是满足 $g^l=k$ 的一个正整数，由循环群和循环子群的讨论，有

$$\text{ord}_{p^e} k = \varphi(p^e)/gcd(l, \varphi(p^e))$$

$$\text{ord}_p k = \varphi(p)/gcd(l, \varphi(p))$$

因为 $p-1$ 和 p^{e-1} 互素，则

$$gcd(l, \varphi(p^e)) = gcd(l, (p-1)p^{e-1}) = gcd(l, p-1) \cdot gcd(l, p^{e-1})$$

因此，

$$\text{ord}_{p^e} k / \text{ord}_p = \frac{\varphi(p^e)/gcd(l, \varphi(p^e))}{\varphi(p)/gcd(l, \varphi(p))} = \frac{(p-1)p^{e-1} \cdot gcd(l, \varphi(p))}{(p-1) \cdot gcd(l, \varphi(p^e))}$$

$$= \frac{p^{e-1}}{gcd(l, p^{e-1})}$$

这就证明了引理。 ♣

定理 n的素性的一个强证据也是一个欧拉证据。

证明 首先，如果n是素数，则它将通过索洛维-斯特拉森或米勒-罗宾检验。

所以我们来考虑奇合数n，设b是一个随机选取的整数，令$c = b^m$，n的素因子分解式为：

$$n = p_1^{e_1} \cdots p_N^{e_N}$$

令b是n的一个强证据，且$c = b^d$。则b是n的一个强证据的假设即意味着：

1) $c = 1 \bmod n$

2) 对某个$1 \leqslant t < s$，$c^{2^t} = -1 \bmod n$

因为对每个整除n的素数方幂p^e，$c^{2^t} = -1 \bmod p^e$，所以$\text{ord}_{p^e} c = 2^{t+1}$。由此还可得出

$$\text{ord}_{p^e} b = 2^{t+1} \times （奇数）$$

（在$c = 1$的情况下，这些阶为奇数，2^0整除他们。）

由引理得$\text{ord}_p b = 2^{t+1} \times (奇数)$，因为惟一可能改变的就是$p$的方幂。

令2^{t_i}为整除$p_i - 1$的2的方幂，则对所有i，$t+1 \leqslant k_i$。设g_i是模p_i的一个本原根，记$b = g_i^{l_i}$，$l_i \mid p_i - 1$，则由循环子群的讨论得

$$\text{ord}_{p_i} b = \frac{p_i - 1}{l_i}$$

因此，对于一个给定的指标i，如果$t+1 < k_i$，则b就是一个模p_i的平方。只有对$t+1 = k_i$，b不是模p_i的平方。

令M为使得b是模p_i的非平方的指标i的个数，且$t+1 = k_i$，则

$$\left(\frac{b}{n}\right)_2 = \left(\frac{b}{p_1}\right)_2^{e_1} \cdots \left(\frac{b}{p_N}\right)_2^{e_N} = (-1)^M$$

另一方面，因为对每个指标i，$2^{t+1} \mid p_i - 1$，我们记$p_i = 1 + 2^{t+1} x_i$，这里x_i为奇数，则

$$p_i^2 = 1 + 2 \cdot 2^{t+1} x_i + 2^{2t+2} x^2 = 1 \bmod 2^{t+2}$$

因此，模2^{t+2}后，所有n中指数大于等于2的所有素数幂正好为1。并且在模2^{t+2}后，

$$p_i = 1 + 2^{t+1} \times (奇数) \equiv 1 + 2^{t+1} \bmod 2^{t+2}$$

因此，n中出现的指数为1的素数幂为$1 + 2^{t+1}$模2^{t+2}提供了因子。由此$\bmod 2^{t+2}$得

$$n = p_1^{e_1} \cdots p_N^{e_N} = (1 + 2^{t+1})^M \bmod 2^{t+2}$$

$$= 1 + \binom{M}{1} 2^{t+1} + \binom{M}{2} 2^{2t+2} + \binom{M}{3} 2^{3t+3} + \cdots = 1 + M \cdot 2^{t+1} \bmod 2^{t+2}$$

这样，n模2^{t+2}就依赖于M是奇数还是偶数：

$$M 是奇数: n = 1 + M \cdot 2^{t+1} = 1 + 2^{t+1} \bmod 2^{t+2}$$

$$M 是偶数: n = 1 + M \cdot 2^{t+1} = 1 \bmod 2^{t+2}$$

因此，在M是奇数的情况下，整除$n-1$的2的方幂正好是2^{t+1}，即$s = t+1$，由此

$$b^{(n-1)/2} = b^{m \cdot 2^{s-1}} = c^{2^{s-1}} = c^{2^t} = -1 \bmod n$$

因为对 M 是奇数，$b^{(n-1)/2}=-1=\left(\frac{b}{n}\right)_2$。

当 M 是偶数时，整除 $n-1$ 的 2 的方幂至少为 2^{t+2}，即 $s\geqslant t+2$，因而有

$$b^{(n-1)/2}=b^{m\cdot 2^{s-1}}=c^{2^{s-1}}=(c^{2^t})^{2^{s-1}-2^t}=(-1)^{2^{s-1}-2^t}=1 \bmod n$$

我们得到 $b^{(n-1)/2}=1=\left(\frac{b}{n}\right)_2$。这就证明了定理，即 n 的素性的强证据也是欧拉证据。 ♣

定理 如果奇数 n 是合数，则至少有 3/4 的整数 b 是 n 的合数性的强（米勒-罗宾）证据，这里 $1<b<n$。

证明 设 k 是满足至少有一个 b 使得 $b^{2^k}=-1 \bmod n$ 的最大非负整数。因为 $(-1)^{2^0}=-1$，所以存在这样的 k。但需要如下引理：

引理 $n=1 \bmod 2^{k+1}$。

证明 由 $b^{2^k}=-1 \bmod n$，当然有 $b^{2^{k+1}}=1 \bmod n$。因此 $n \mid b^{2^{k+1}}-1$，进而由费马观察，对任何整除 n 的素数 p 以及对某个 $l<k+1$，或者 $p \mid b^{2^l}-1$，或者 $p=1 \bmod 2^{k+1}$。因为由假设 $b^{2^k}=-1 \bmod n$，而对 $r<k$，$b^{2^r} \bmod n$ 既不为 1 也不为 -1。因此不可能有 $p \mid b^{2^l}-1$，对某个 $l<k+1$。故任何整除 n 的素数 p 满足 $p=1 \bmod 2^{k+1}$，那么将任意多个这样的素数乘起来构成 n，仍然是模 2^{k+1} 同余 1，即 $p=1 \bmod 2^{k+1}$。 ♣

现在我们回到定理的证明上。设 $l=2^k \cdot m$，且 $n-1=2^s \cdot m$，这里 m 为奇数。由引理，$2l \mid n-1$。定义 $G=\mathbb{Z}/n^{\times}$ 的子群如下：

$$H=\{g\in G: g^{n-1}=1 \bmod n\}$$

$$I=\{g\in G: g^l=\pm 1 \bmod p_i^{e_i}, \text{对所有} i\}$$

$$J=\{g\in G: g^l=\pm 1 \bmod n\}\supset\{\text{强说谎者}\}$$

$$K=\{g\in G: g^l=1 \bmod n\}$$

不难验证这四个子群有如下包含关系：（由子群的定义也容易验证它们是 G 的子群）

$$G\supset H\supset I\supset J\supset K$$

引理 强说谎者（n 的素性的错误证据）集合包含于子群 J 中。

证明 首先，如果 $b^m=1 \bmod n$，则由 $m \mid l$ 可知，必有 $b^l=1 \bmod n$。另一方面，如果 $b^{m\cdot 2^t}=-1 \bmod n$ 对某个 $t<s$ 成立，则由 k 的定义 $t\leqslant k$，因此

$$b^l=b^{m\cdot 2^k}=(b^{m\cdot 2^t})^{2^{k-t}}=(-1)^{2^{k-t}} \bmod n$$

所以任何说谎者都在 J 中。 ♣

接下来，除了 $n=9$ 的情况比较容易不予讨论外，我们将证明 $[G:J]\geqslant 4$。因为强说谎者都包含于 J，这将证明 n 的合数性证据的个数将至少为

$$(n-1)-\frac{1}{4}\varphi(n)\geqslant(n-1)-\frac{1}{4}(n-1)=\frac{3}{4}(n-1)$$

令 $f: G\to G$ 由映射 $f(g)=g^l$ 所定义，因为 G 是交换的，所以该映射是一个群同态。

引理 设 $S=\{a\in G: a=\pm 1 \bmod p_i^{e_i}, \text{对所有的} i\}$，$S$ 的每个元素是 G 中某个元素的 2^k 次方幂。因此，S 的每个元素是 G 中某个元素的 l 次方幂。即群同态 $f: G\to S$ 是满射。

证明 设 x 为一整数，使得 $x=b \bmod p_i^{e_i}$ 或者 $x=b^2 \bmod p_i^{e_i}$，不同的素数 p_i 有不同的选择，b 则是上述的特定元素。由此如果 $x=b^2 \bmod p_i^{e_i}$，则有 $x^{2^k}=+1 \bmod p_i^{e_i}$；如果 $x=b \bmod p_i^{e_i}$，$x^{2^k}=-1 \bmod p_i^{e_i}$。这就证明了引理的第一个论断。

因为 m 为奇数，$\pm 1 \bmod p_i^{e_i}$ 都是自己的 m 次方幂。因此，如果 $g^{2^k}=h$，$h\in S$，则对所有指标 i，

$$g^I = g^{2^k \cdot m} = h^m = h \bmod p_i^{e_i}$$

这就证明了 f 为满射。 ♣

我们还希望证明 K 在 I 中的指标为 2^N。我们将以一个在群中运用更加普遍的一个引理的推论形式证明这一结论。

引理 设 X,Y 为有限群，$h: X \to Y$ 为群同态，Z,W 是 Y 的子群，且 $Z \supset W$，并假设 Z 包含于 f 的像 $f(X)$ 中，令

$$h^{-1}(Z) = \{g \in G : h(g) \in Z\}$$
$$h^{-1}(W) = \{g \in G : h(g) \in W\}$$

则有如下指标公式成立：$[Z:W] = [h^{-1}(Z) : h^{-1}(W)]$。

证明 令 V 是群同态 $h: X \to Y$ 的核，我们将要证明

$$|h^{-1}(Z)| = |V| \cdot |Z| \text{ 和 } |h^{-1}(W)| = |V| \cdot |W|$$

这样我们就得出引理的结论：

$$[Z:W] = \frac{|Z|}{|W|} = \frac{|V| \cdot |Z|}{|V| \cdot |W|} = \frac{|h^{-1}(Z)|}{|h^{-1}(W)|} = [h^{-1}(Z) : h^{-1}(W)]$$

Z 的原像 $h^{-1}(Z)$ 是 $z \in Z$ 的原像 $h^{-1}(z) = \{x \in X : h(x) = z\}$ 的不相交的并集。如果能够证明 $h^{-1}(z)$ 的元素个数等于 $|V|$，则可得到

$$|h^{-1}(Z)| = \text{集合} h^{-1}(z) \text{的基数之和，} z \in Z = |Z| \cdot |V|$$

这是因为共有 $|Z|$ 个不同的集合 $h^{-1}(z)$，而且我们希望每个这样的集合的基数为 $|V|$。

因此问题就简化为证明对任一 $z \in Z$, $h^{-1}(z)$ 的元素个数等于 $|V|$。为证明这个结论，我们构造一个双射如下：

$$b: V \to h^{-1}(z)$$

这样只需要证明集合有相同的元素个数，避免直接计数。因为 z 在 h 的像中，我们至少可以找到一个 $x_0 \in X$，使 $h(x_0) = z$，那么可以尝试将映射 b 定义为：$b(v) = v \cdot x_0$。首先我们必须验证这个映射是 V 到 $h^{-1}(z)$ 的映射，即必须验证对所有的 $v \in V$，$h(b(v)) = z$。实际上，因为对 $v \in V$，V 是 h 的核，显然有

$$h(b(v)) = h(v \cdot x_0) = h(v) \cdot h(x_0) = e_Y \cdot z = z$$

其次，我们需要验证 b 是一个单射，假设对 $v, v' \in V$，有 $b(v) = b(v')$，我们来验证 $v = v'$。由映射 b 的定义，$v \cdot x_0 = v' \cdot x_0$，两边同时右乘 x_0^{-1}，即得 $v = v'$，因此 b 是一个单射。

最后，我们还需要验证 b 是一个满射。给定 $q \in h^{-1}(z)$，要寻找元素 $v \in V$，使 $q = b(v)$。我们来验证 $q \cdot x_0^{-1} \in V$：

$$h(q \cdot x_0^{-1}) = h(q) \cdot h(x_0^{-1}) = z \cdot h(x_0)^{-1} = z \cdot z^{-1} = e_z$$

这就证明了 $b: V \to h^{-1}(z)$ 是一个双射，也就证明了 $h^{-1}(z)$ 的元素个数等于 $|V|$。因此，引理得证明。 ♣

推论 $[I:K] = [f^{-1}(S) : f^{-1}(\{e\})] = [S : \{e\}] = 2^N$

证明 由 f 的定义，$f(g) = g^I$，$K = f^{-1}(\{e\})$。上面证明第三个引理表明，$I = f^{-1}(S)$，再由刚刚证明的引理，$[f^{-1}(S) : f^{-1}(\{e\})] = [S : \{e\}]$。因为对整除 n 的 N 个不同的素因子 pi，S 包含 ± 1 两种选择，所以 $|S| = 2^N$。这就证明了推论。 ♣

利用刚才证明的引理，我们还有如下推论：

推论 令 $P=\{\pm 1 \bmod n\}$，$E=\{e_G\}$，则 $[J:K]=[f^{-1}(P):f^{-1}(E)]=[P:E]=2$。

这样我们已经得到 $I \supset J \supset K$，$[I:K]=2^N$ 以及 $[J:K]=2$，而在先前群指标的讨论中，我们也证明如下的乘法性质：

$$[I:J]\cdot[J:K]=[I:K]$$

因此，$[I:J]=[I:K]/[J:K]=2^{N-1}$。又因为 $[G:J]=[G:I]\cdot[I:J]$，所以有

$$[G:J]\geqslant[I:J]=2^{N-1}$$

因为强（米勒-罗宾）说谎者均包含于 J，我们有 $\dfrac{\text{说谎者的数量}}{|G|}\leqslant\dfrac{1}{2^{N-1}}$。如果 n 的不同素因子的个数至少为 3，即 $N\geqslant 3$，则

$$\frac{\text{说谎者的数量}}{|G|}\leqslant\frac{1}{4}$$

如果不同素因子的个数 N 为 2，则我们知道 n 肯定不是卡米克尔数，也就是前面的群 H 是 $G=\mathbf{Z}/n^{\times}$ 的一个真子群。由拉格朗日定理，$[G:H]\geqslant 2$，利用子群指标的乘法性质有：

$$[G:J]=[G:H]\cdot[H:I]\cdot[I:J]\geqslant[G:H]\cdot[I:J]\geqslant 2\cdot 2^1=4$$

因此，我们断定说谎者在 G 所有的元素中所占比例小于 1/4。

最后，假设 $n=p^e$，即一个素数方幂的情形（即 $N=1$）。此时由本原根的存在性，$\mathbf{Z}/p^{e\times}$ 是循环群，群 H 则为

$$H=\{g\in\mathbf{Z}/p^{e\times}:g^{p^e-1}=1 \bmod p^e\}$$

由循环群的讨论，要确定 $|J|$，我们可以利用加法群 $\mathbf{Z}/\varphi(p^e)$ 与乘法群 $\mathbf{Z}/p^{e\times}$ 的同构。这样，在加法群上讨论，需要知道如下方程的解的个数：

$$(p^e-1)\cdot x=0 \bmod \varphi(p^e)$$

即 $(p^e-1)\cdot x=0 \bmod (p-1)p^{e-1}$。这样的方程我们可以求解，去掉公因子则变形为

$$\frac{p^e-1}{p-1}\cdot x=0 \bmod p^{e-1}$$

因为系数 x 与模数互素，因此它有乘法逆元，所以方程就等价于 $x=0 \bmod p^{e-1}$。因为 x 是模 p^e 的整数，我们恰好得到 $p-1$ 个模 $\varphi(p^e)$ 的不同解。

因此，

$$[G:J]=[G:H]\cdot[H:J]\geqslant[G:H]=\frac{\varphi(p^e)}{p-1}=p^{e-1}$$

除了 $p=3$，$e=2$ 的情况，我们均得到 $[G:J]\geqslant 4$。由此，我们断言至多有 1/4 的元素是说谎者。

对 $n=9$ 的特殊情况，正好有两个强的说谎者 ± 1，且 $2/(9-1)=1/4$。

这就证明了定理，也表明米勒-罗宾的检验方法可以运行。 ♣

第24章 因式分解攻击

与检验一个大整数否为素数相比，尝试分解一个大整数则是一个更加困难的问题，而证明一个伪素数是真正的素数比验证它的伪素性更为困难。

不论如何，在使用任何成熟的检验方法之前，先用较小的素数做一定数量的除法试验，从效率的观点看这无疑是明智的举措。

在尝试一个整数n之前，首先检查它是否能够通过拟素性检验也是明智的。因为即使成熟的分解方法也有失效的可能，比如碰巧n是素数，由此造成的失效却是难以识别的。也就是说，在尝试分解n的任何失败中，要弄清楚究竟是在算法执行中的运气不好，还是n根本就是素数，这一点相当重要，如果运气差还可以做一点校正并重新执行算法。然而，多数成熟的分解方法在素性检验方面却表现不佳，这归咎于它们的失效模式。因此我们只讨论已知大整数是合数（由费马或米勒-罗宾检验）的分解。

同样，素性证明方法也应当只在已知一个数为伪素数（即一个**可能的素数**）时应用。

从实用的角度出发，我们假设所有“小”素因子已经被去掉了，可能用试除法，这一点无论什么时候都是有益的。（至于“小”的准确含义可依赖上下文来理解。）

我们这里介绍的两种分解方法，并不代表这一问题的现状，主要是因为它们容易解释、理解和执行。而 Rho 方法还创造性地利用了“生日悖论”的思想。$p-1$方法是很重要的，因为在公钥密码体制中所用素数的选取，必须能避免这种分解和相关的攻击。

24.1 Pollard 的 Rho 方法

Pollard 的 Rho 方法可以较快地找到合数的较小因子。这是一个比较简单的分解方法，它在分解素因子在 1 000 000 附近的整数时，速度要比试除法快好几倍。一方面，该方法确有实际的效果，如果一个数有小的素因子，则它能以比寻找一个大因子更快的速度找到这个小因子。另一方面，它有一个较小的不足，就是它是概率意义上的方法，所以必须对“坏情况”的出现有足够的警惕。除此之外，该方法有一个特殊的不足之处，即它难以证明它的执行效果：一方面，从试验角度看，这可能被认为是没有结果的。另一方面，在比较重视正确性的应用中，这显然是一个致命的缺陷。该方法第一次出现在文献中[Pollard 1975]。

该算法的描述比较简单，给定整数n，设初始值$x=2$，$y=x^2+1$：

- 计算$g=gcd(x-y,n)$；
- 如果$1<g<n$，则停止：g就是n的一个因子；
- 如果$g=1$，用x^2+1代替x，$(y^2+1)^2+1$代替y，返回到第一步；
- 如果$g=n$，失败，需要对算法重新初始化，但这种情况很少出现。

备注 利用生日悖论，找到n的一个因子p需要的循环数应当约为$\sqrt{p}$阶的。如果n为合数，则存在一个素因子$p\leqslant\sqrt{n}$。所以该算法找到一个真因子需要$\sqrt[4]{n}$阶的循环（以下进行注释）。

备注 如上所述，我们不能对算法运行的循环数强加一个限制。如果已知 n 为合数，这是最可能接受的，因为可能出现的最坏情况就是算法需要的时间与试除法一样多。最成问题的情况是 $g=n$，所幸的是这种情况出现的概率极小（难以估计）。

备注 在该算法的执行过程中，对执行的循环数应当加以可能的限制，当然是在校正之前，因为这个限制有几种选择。$\sqrt{n}$ 步太多，而且无论如何它也不比试除法来得快。$\sqrt[4]{n}$ 步不足以允许"坏情况"的出现。在实际执行中，$100\sqrt[4]{n}$ 个循环似乎对找到一个真因子是足够的。尽管如此，对一个已知的合数 n，让该算法要么执行成功，要么失效。这是十分合理的，这不仅是因为算法要运行快速，没有失效，而且是因为对算法执行的循环数给出一个严格估计而加以恰当的限制似乎很困难。有关 Pollard 的 Rho 方法的一个严格的结果，可参见[Bach 1991]。

备注 同前面提到的一样，因为该算法最坏的情况就是完全失效，故用此算法来检验素性是不明智的，因为对合数它也可能出现失效。因此，在实际执行 Pollard 的 Rho 算法之前，首先运用对 n 的素性检验。这里可以运用费马检验，因为 n 没有通过费马检验，则 n 确定地为一个合数。如果该法不行，还可以运用米勒-罗宾检验。

备注 在 H.Cohen 的《*A Course in Computational Algebraic Number Theory*》（Springer-Verlag,1993）一书中，作者认为应付"失效"状态的最好办法是使用一个不同于 $x\to x^2+1$ 的函数。尽管 $c=0,-2$ 是坏情况，$f(y)=y^2+c$ 计算起来最简单，可以作为一种选择。该书还断言，简单地改变 x 的初值，即从上述的 $x=2$ 变为其他值，不是聪明之举。

备注 尽管在实践中倾向于把由该方法找到的真因子当作素因子，但没有任何东西保证这个真因子 g 就是素数。然而，寻找一个非素数的真因子却是分解过程中完全有用的步骤。

该算法为什么可行？很不幸的是，最直接也最容易理解的解释根本就是不严格的，而且没有找到更加严格的解释，也许这正是该算法的迷人之处。无论如何，有两个关键的部分：即包含在生日悖论中的概率思想和 Floyd 循环检测方法，而 Floyd 循环检测方法则是生日悖论思想得以应用的基础。

假设 n 为一个正整数，d 是它的真因子，d 是否为素数并不重要，只是 d 与 n 相比是非常小的。由生日悖论，我们知道如果存在超过 $\sqrt{d}$ 个整数 $x_1,x_2,\cdots,x_t$，则这些数中有两个数模 d 相等的概率将大于 1/2。Pollard 的 Rho 方法的思想则是 $\sqrt{d}$ 远小于 $\sqrt{n}$，所以我们期望，如果我们选择一个整数"随机序列" $x_1,x_2,\cdots$，那么在找到两个整数模 n 相等之前，就已找到两个数模 d 相等了。也就是说，假设 $x_i=x_k \bmod d$，但还不知道 d 是什么，我们可以通过计算求得 d（注意，用欧几里得算法计算 gcd 是相对容易的）：

$$g=gcd(x_i-x_j,n)$$

更准确一点，如果 $x_i=x_k \bmod d$，但 $x_i\neq x_k \bmod n$，则这个 gcd 将是 d 的倍数，而且是 n 的一个因子，严格小于 n 本身。这也被认为是分解 n 的有用步骤。

一种过于天真的实现方法就是对所有的 $i<j$，利用计算最大公因子 $gcd(x_i-x_j,n)$ 来寻找 n 的真因子。这是一个糟糕的主意：如果为了寻找真因子 d 我们希望用大约 $\sqrt{d}$ 个随机整数，那么将不得不进行 $\frac{1}{2}(\sqrt{d})^2=\frac{1}{2}d$ 阶的 gcd 计算。也就是说，我们在用试除法求 d（试除法的每个步骤都要比欧几里得算法简单）。

所以我们需要一种聪明的方法来利用生日悖论。也就是在这里我们利用了 Pollard 的 Rho

方法产生假定随机整数的特定方式。首先，我们假设算法中使用的函数 $f(x)=(x^2+1)\%n$ 是 $\mathbf{Z}/n$ 到本身的一个**随机映射**，因为在这方面也没有人给出更多事实的证明，我们也不想确切地说明这里的含义是什么。但不论如何，如果我们用如下的方式构造一个假设的随机数序列，则序列中的后一个元素都完全决定于前一个元素：

$$x_0=2$$
$$x_1=f(x_0)$$
$$x_2=f(x_1)$$
$$x_3=f(x_2)$$
$$\cdots$$

也就是说，如果对 $i<j$ 有 $x_j=x_i \bmod d$，则必有

$$x_{j+1}=x_{i+1}, x_{j+2}=x_{i+2}, x_{j+3}=x_{i+3} \quad \bmod d$$

等等。特别地，对所有 $t\geqslant j$，$x_t=x_{t-(j-i)} \bmod d$。同理，只要 $t-l(j-i)\geqslant i$，就有

$$x_t=x_{t-(j-i)}=x_{t-2(j-i)}=x_{t-3(j-i)}=\cdots=x_{t-l(j-i)} \bmod d$$

即比起我们仅有的一个递增的随机数集合，这里还有一个更加结构化的东西。

注意我们这里不必关注第一个 $x_i=x_j$ 的情况，但却要关注相对早的情况。**Floyd 循环检测方法**就是寻找匹配的有效方法。首先，我们不必记住整个 x_i 的列表，这样做效率太低。我们只要记住最后一个，同时还要计算一个序列：

$$y_1=x_2$$
$$y_2=x_4$$
$$y_3=x_6$$
$$y_4=x_8$$
$$\cdots$$
$$y_i=x_{2i}$$

计算 y_i 序列的高效方法是利用下面这个公式，并且只需要记住最后一个 y_i。

$$y_{i+1}=(y_i^2+1)^2+1\%n$$

在每一步，我们只记住了计算所得的 x_t 和 y_t，并且考虑 $gcd(x_t-y_t,n)$。为什么这最有可能找到一个真因子呢？设 $i<j$，j 是第一个满足 $x_j=x_i \bmod d$ 的下标，利用前面的讨论，这意味着只要 $t-l(j-i)\geqslant i$，有

$$x_t=x_{t-l(j-i)} \bmod d$$

所以取 $t=2s$：对所有满足 $t-l(j-i)\geqslant i$ 的 s，

$$y_s=x_{2s}=x_{2s-l(j-i)} \bmod d$$

所以当 $s\geqslant i$，$s=l(j-i)$ 且 $l(j-i)\geqslant i$（对足够大的 l 亦成立）时，

$$y_s=x_{2s}=x_{2s-l(j-i)}=x_{2s-s}=x_s \bmod d$$

这就证明了上述所用的方法在求 x_i, x_j 使得 $x_j=x_i \bmod d$ 时真正有效，假设它们存在的话。

备注 注意在后面比较冗长的讨论中，假设了函数 $x\to x^2+1$ 的动作是“随机”的，而我们对此没有提供任何保证。而且，假设了这个映射的随机性，才有可能在一个匹配情况出现之前对期望的循环数做出一个更加准确的分析。这个分析与我们上面的讨论有关，而且较之更加成熟。

习题

24.1.01 用 Pollard 的 Rho 方法求 1133 的一个真因子。

24.1.02 用 Pollard 的 Rho 方法求 1313 的一个真因子。

24.1.03 用 Pollard 的 Rho 方法求 1649 的一个真因子。

24.2 Pollard 的 $p-1$ 方法

这个方法有点特殊，它只能应用在求一个素因子 p，使 $p-1$ 能被“小”因子整除的情况下，除此之外该方法将无法正常应用。但是这个检验的运用是相当简单的，所以在防止因子分解攻击时，必须考虑这一方法。

给定一个整数 B，一个整数 n 称为是 B **平滑**的，如果它的所有素因子均小于等于 B。在本节中，B 就是**平滑性界限**。一个整数 n 称为是 B **方幂平滑**的，如果它的所有素数方幂因子小于等于 B。

给定一个整数 n，Pollard 的 $p-1$ 算法寻找 n 的一个 B 平滑的素因子 p，将需要 $O(B\ln n/\ln B)$ 次模 n 乘法。

备注 保持 B 较小显然是我们所期望的，它应当是小于 $\sqrt{n}$，或者通过做试除法一无所获。另一方面，B 的值太小将导致算法不能找到因子，在实践中，因为 B 的值必须保持较小而不能找到所有的因子，这个算法正好用在我们运气较好且某个 $p-1$ 有全部较小的素因子的情况，这里的 p 就是我们讨论大数的一个素因子。

备注 一个随机选取的数，我们希望它的素因子是多大呢？这个问题非同一般，并且没有一个简单答案。

给定一个已知是合数的整数 n，**但不是一个素数的一个方幂**，并且给出一个平滑性界限 B，选取一个随机整数 b，$2\leqslant b\leqslant n-1$。计算 $g=gcd(b,n)$，如果 $g\geqslant 2$，则 g 就是一个真因子，停止；否则，令 $p_1,p_2,\cdots,p_t$ 为小于等于 B 的素数，对 $i=1,2,3,\cdots,t$，令 $q=p_i$，并且

- 计算 $l=\text{floor}(\ln n/\ln p_i)$；
- 用 b^{q^l} 代替 b；
- 计算 $g=gcd(b-1,n)$；
- 如果 $1<g<n$：结束，g 就是一个真因子；
- 如果 $g=1$：继续；
- 如果 $g=n$：结束，失败。

备注 如果 B 相当大，则可以节约一些时间，不用在一开始就计算所有的 $p_1,\cdots,p_t$，只需要计算所需要的，并希望之前就找到一个因子。另一方面，如果正在检验几个数，则推迟计算所有的素数列表是没有意义的。

$p-1$ **方法为什么可行？**如果我们可以对某个整数 b，得到 $1<gcd(b-1,n)<n$，则已经找到了一个真因子 $g=gcd(b-1,n)$。在该算法中，令

$$p=1+p_1^{e_1}\cdots p_t^{e_t}$$

是 n 的一个 B 平滑的素因子，e_i 为整数指数。对任意与 p 互素的整数 b，由费马小定理，$b^{p-1}=1$，即

$$b^{p_1^{e_1}\cdots p_t^{e_t}}=1\bmod p$$

而 $l_i = \text{floor}(\ln n / \ln p_i)$ 大于或等于 e_i。令 $T = p_1^{l_1} \cdots p_t^{l_t}$，则 $p_1^{e_1} \cdots p_t^{e_t}$ 整除 T。因此，$b^T = 1 \bmod p$，这里的整数 b 与 p 互素。即 $p \mid gcd(b^T - 1, n)$。

注意，上面给出的实际算法要比前面这一段中指出的方法更频繁地计算 gcd，这也就避免了 $gcd(b^T - 1, n) = n$ 的可能。

该算法可能出现失效的原因有两方面。一是 n 没有使得 $p-1$ 是 B 平滑的素因子 p，在这种情况，计算出的 gcd 总是1。另一种失效模式是 n 的所有素因子 q 都使得 $q-1$ 是 B 平滑的，此时 $gcd(b^T - 1, n) = n$。也就是这个原因，上面给出的算法还要计算额外的 gcd，即使所有计算所得的 gcd 为1或 n，只要在这些 gcd 中 n 至少出现一次，则算法就有成功的希望：只是简单地换一个不同的初始随机数 b 重新开始。

例子 用 $p-1$ 方法分解 $9991 = 97 \cdot 103$。注意 $97 - 1 = 2^5 \cdot 3$，所以 $97-1$ 是 {2,3} 平滑的，取初始值 $b = 3$，$\log_2 9991$ 的整数部分为13，因此首先计算

$$b^{2^{13}} \%9991 = 229$$

利用欧几里得算法，计算得 $gcd(9991, 229 - 1) = 1$，所以在这一轮求 9991 的因子失败。接下来，$\log_3 9991$ 的整数部分为 8。故计算 $b^{3^8} \%9991$，即 $b^{3^8} \%9991 = 3202$，利用欧几里得算法，计算出 $gcd(9991, 3202 - 1) = 97$，因此就找到了一个因子 97。

例子 分解 $3801911 = 1801 \cdot 2111$，因子 1801 为 {2,3,5} 平滑的：$1801 - 1 = 2^3 \cdot 3^2 \cdot 5^2$。取初始值 $b = 3$，$\log_2 3801911$ 的整数部分为 21，所以首先计算

$$b^{2^{21}} \%3801911 = 3165492$$

利用欧几里得算法，$gcd(3801911, 3165492 - 1) = 1$，所以这一轮寻找 3801911 的因子失败。再来看 $p_2 = 3$ 的情况，$\log_3 3801911$ 的整数部分为 13，所以计算

$$b^{3^{13}} \%3801911 = 2431606$$

利用欧几里得算法求得

$$gcd(3801911, 2431606 - 1) = 1$$

这一轮又告失败。最后，来看 $p_3 = 5$，$\log_5 3801911$ 的整数部分为 9，所以计算

$$b^{5^9} \%3801911 = 2604247$$

利用欧几里得算法求得

$$gcd(3801911, 2604247 - 1) = 1801$$

因此，我们找到了 3801911 的一个因子 1801。

备注 实际上，在中间过程可以跳过对 gcd 的计算，这些中间计算仅仅是为了避免某些相对不可能的重合情况。

例子 分解 $54541557732143 = 54001 \cdot 1010010143$。素数 54001 是 {2,3,5} 平滑的，初始值 $b = 3$。这一次我们跳过中间 gcd 的计算。2 的指数为 45，3 的指数为 28，5 的指数为 19，计算

$$b^{2^{45}} \%54541557732143 = 1333741139152$$
$$b^{3^{28}} \%54541557732143 = 22167690980770$$
$$b^{5^{19}} \%54541557732143 = 2268486536233$$

最后计算

$$gcd(54541557732143, 2268486536233 - 1) = 54001$$

因此，我们就找到了 54541557732143 的一个因子 54001。

习题

24.2.01 利用 Pollard 的 $p-1$ 方法求 901 的一个素因子 p，使得 $p-1$ 为 B 平滑的，这里的因子基 $B=\{2\}$。

24.2.02 利用因子基 $B=\{2,3\}$ 的 Pollard 的 $p-1$ 方法，求 11009 的一个素因子 p，使得 $p-1$ 为 B 平滑的。

24.2.03 利用因子基 $B=\{2,3\}$ 的 Pollard 的 $p-1$ 方法，求 11227 的一个素因子 p，使得 $p-1$ 为 B 平滑的。

24.2.04 利用因子基 $B=\{2,3\}$ 的 Pollard 的 $p-1$ 方法，求 11663 的一个素因子 p，使得 $p-1$ 为 B 平滑的。

24.3 Pocklington-Lehmer 准则

这是一种专门的技术，特别适用于整数 N 的素性检验，使得 $N-1$ 可以分解。这种方法最常用于特殊形式的整数，比如费马数 $2^{2^n}+1$。该方法的一个更为成熟的变形给出了对梅森数 2^n-1 素性的 Lucas-Lehmer 检验，这些数中有目前已知的最大素数。该检验不是概率意义上的，所以它在适用的时候给出了素性的证明。这一思想源自于 1876 年和 1891 年 Eduard Lucas 的工作，费马小定理的一种真逆命题。

这一技术可用于证明整数 N 的素性，或减少真因子的搜索空间，条件是 $N-1$ 的分解是已知的。

引理 设 N 是一个正整数，q 是 $N-1$ 的一个素因子。设 b 是满足 $b^{N-1}=1 \bmod N$，但 $gcd(b^{(N-1)/q}-1,N)=1$ 的整数，令 q^e 是恰好整除 $N-1$ 的 q 的方幂，则 N 的任意正因子 d 满足 $d=1 \bmod q^e$。

证明 设 d 是 N 的一个正因子，因为我们可以将 d 表示为它的素因子的乘积，即 $d=\prod_i p_i^{e_i}$，所以如果能够证明 $p_i=1 \bmod q^e$，则自然有 $d=1 \bmod q^e$。因此考虑 N 的素因子 $p=d$ 的情况就足够了。

引理的一个假设条件还给出了这样一个事实，即 $b\cdot b^{N-2}=1 \bmod N$，因此 b 是 b^{N-2} 模 N 的乘法逆元，所以它与 N 互素，这样 b 与 p 也就是互素的。

令 t 是 b 在乘法群 $\mathbf{Z}/p^{\times}$ 中的阶，即 $b^t=1 \bmod p$，但不存在比 t 更小的整数满足这一条件。而由费马小定理可知，$b^{p-1}=1 \bmod p$，所以 $t \mid p-1$。另一方面，因为

$$gcd(b^{(N-1)/q}-1,N)=1$$

当然有

$$gcd(b^{(N-1)/q}-1,p)=1$$

因此 p 不能整除 $b^{(N-1)/q}-1$，即 $b^{(N-1)/q}\neq 1 \bmod p$。这也就是说，$t$ 不能整除 $(N-1)/q$。又因为 $b^{N-1}=1 \bmod N$，当然有 $b^{N-1}=1 \bmod p$，所以 $t \mid N-1$。

那么由 $t \mid N-1$ 且 t 不能整除 $(N-1)/q$ 可得 $q^e \mid t$，又因为 $t \mid p-1$，则 $q^e \mid p-1$，即 $p=1 \bmod q^e$。这就证明了我们的结论。 ♣

定理 设 $N-1=K\cdot U$，这里 K 和 U 为互素的整数，K 的分解式是已知的，并且 $K>\sqrt{N}$，则

- 如果对每个整除 K 的素数 q，存在一个 b，使得 $b^{N-1}=1 \bmod N$，但是 N 与 $b^{(N-1)/q}-1$ 互素，即 $gcd(b^{(N-1)/q}-1,N)=1$，则 N 为素数。
- 如果 N 为素数，则对每个整除 K 的素数 q，存在一个 b，使得 $b^{N-1}=1 \bmod N$，但是 N 与 $b^{(N-1)/q}-1$ 互素，即 $gcd(b^{(N-1)/q}-1,N)=1$。

证明　如果 N 为素数，则 $\mathbf{Z}/N$ 有一个本原根 b，它满足 $b^{N-1}=1 \bmod N$，但 $gcd(b^{(N-1)/q}-1,N)=1$。

现在假设存在那样一个 b 满足所述条件，我们来证明 N 为素数。假设每个整除 K 的素数 q，存在一个 b，使得 $b^{N-1}=1 \bmod N$，但是 N 与 $b^{(N-1)/q}-1$ 互素，则由引理，N 的所有因子模 K 同余 1。如果 N 不是素数，则它有一个（素）因子 d，$1<d<\sqrt{N}$。但是 $d=1 \bmod K$，$K>\sqrt{N}$。这就出现了矛盾，因此 N 是素数。 ♣

这个定理对于检验具有特殊形式的整数的素性，给出了一个非常有效的方法。

推论[Proth 1878]　设 $N=u2^n+1$，这里 u 为奇数且 $u<2^n$。假设存在一个 b，使 $b^{(N-1)/2}=-1 \bmod N$，则 N 为素数。

证明　由已知条件有 $N-1=2^n \cdot u$，因为 $u<2^n$，$2^n>\sqrt{N}$，将这里的条件用到前面定理中，即 $K=2^n$，$U=u$，而 $b^{(N-1)/2}=-1 \bmod N$ 即是指 $b^{(N-1)/2}-1=-2 \bmod N$，因此 $gcd(b^{(N-1)/2}-1,N) \mid gcd(-2,N)=1$，这是因为 N 为奇数。这样就完全符合上述定理的条件，所以 N 为素数。 ♣

下面的定理给出了进一步改进的可能。注意到看似特别的一个条件：**对某个给定的界 B，未分解部分没有小于等于 B 的素因子**，却是一个非常实用的条件，因为在实践中总是用试除法和 Pollard Rho 方法去掉了 $N-1$ 的小的素因子。

定理　假设 $N-1=K \cdot U$，这里 K 的分解是已知的，$gcd(K,U)=1$，且未分解部分 U 的所有素因子是大于一个界 B 的。假设 $B \cdot K \geqslant \sqrt{N}$，并假设对每个整除 K 的素数 q，存在 b（依赖于 q）使 $b^{N-1}=1 \bmod N$，但 $gcd(b^{(N-1)/q}-1,N)=1$。还假设存在一个 b_0，使 $b_0{}^{N-1}=1 \bmod N$ 且 $gcd(b_0^K-1,N)=1$，则 N 为素数。反之，如果 N 为素数，则满足上述条件。

证明　这个定理证明与前面引理的证明类似。设 p 是 N 的一个素因子，由前面引理，$p=1 \bmod K$。令 t 是 b_0 在 $\mathbf{Z}/p^{\times}$ 中的阶，则由费马小定理，$t \mid p-1$。而由最后的假设，$t \mid N-1$ 且 t 不整除 K。注意 $K=(N-1)/U$，因为 t 不整除 K 且 $t \mid N-1=K \cdot U$，不可以有 t 与 U 互素。由于 U 所有素因子均大于 B，所以 $gcd(t,U)>B$。又因为 $gcd(K,U)=1$，由 $t \mid p-1$ 和 $K \mid p-1$ 以及惟一分解，我们有 $lcm(t,K) \mid p-1$。特别地，$gcd(t,U) \cdot K \mid p-1$。因为 $gcd(t,U)>B$ 且 $K \cdot B>\sqrt{N}$，可以得到 $p-1>B \cdot K>\sqrt{N}$。因此 N 没有小于等于 $\sqrt{N}$ 的素因子，所以它是素数。 ♣

作为这个定理的特殊应用，我们来检验费马数 $N=F_n=2^{2^n}+1$ 的素性。前面的推论给出了素性的一个充分条件，但如果我们利用二次互反律，可以证明费马数在特殊情况下一个充分必要条件。

推论（Pepin 检验）　第 n 个费马数 $F_n=2^{2^n}+1$ 为素数当且仅当

$$3^{(F_n-1)/2}=-1 \bmod F_n$$

证明　这个结论就是前面那个推论中 $k=1$ 的情况。一方面，如果同余式 $3^{(F_n-1)/2}=-1 \bmod F_n$ 成立，且 $F_n=1 \cdot 2^{2^n}+1$，$1<2^{2^n}$，因此 F_n 是素数。

另一方面，假设 F_n 是素数，则我们可以利用二次互反律计算**二次符号**：

$$\left(\frac{3}{F_n}\right)_2 = (-1)^{(3-1)(F_n-1)/4} \cdot \left(\frac{F_n}{3}\right)_2 = \left(\frac{F_n}{3}\right)_2$$

我们再来做模 3 的计算：$F_n = 2^{2^n}+1 = (2^{2^n-1})^2+1$，因为 2^{2^n-1} 模 3 是非零的，即它模 3 为 ±1，所以它的平方为 1。因此，$F_n = 2^{2^n}+1 = (2^{2^n-1})^2+1 = 1+1 = 2 \bmod 3$。因为 2 不是模 3 的平方，所以

$$\left(\frac{F_n}{3}\right)_2 = \left(\frac{2}{3}\right)_2 = -1$$

再由欧拉准则，我们得到

$$\left(\frac{3}{F_n}\right)_2 = 3^{(F_n-1)/2} \bmod F_n$$

即 $3^{(F_n-1)/2} = -1 \bmod F_n$。

因此若 F_n 是素数，利用二次互反律和欧拉准则就得到了推论中的同余式。 ♣

备注 似乎只有最前面 5 个费马数已知是素数：

$$2^{2^0}+1 = 3$$
$$2^{2^1}+1 = 5$$
$$2^{2^2}+1 = 17$$
$$2^{2^3}+1 = 257$$
$$2^{2^4}+1 = 65537$$

这些数费马已认识到是素数，目前已知紧接着的 19 个费马数是合数。费马曾宣称第 5 个费马数 $F_5 = 2^{2^5}+1 = 4294967297$ 也是素数，但在 100 年后，欧拉发现这个数有一个真因子 641：

$$4294967297 = 641 \cdot 6700417$$

欧拉并没有用穷尽的方法，他首次证明并使用了如下引理，从而明显减小了因子搜索空间。他的证明类似于费马对 b^n-1 可能素因子的发现。通过 128 因子获得了一种更加快速的搜索方法。

引理 $2^{2^n}+1$ 的每个素因子 p 满足 $p = 1 \bmod 2^{n+2}$。

我们首先看一些例子，然后再给出引理的证明。

由此，为了寻找 $F_5 = 2^{2^5}+1$ 的因子，我们只需要考查形如 $k \cdot 2^7+1$ 的那些素数：257，641 等等。如果 257 不能整除 $F_5 = 2^{2^5}+1$，容易验证 641 整除 $F_5 = 2^{2^5}+1$。这比起从素数 2、3 开始，逐个试验一直到 641 的 116 个素数要容易得多。

备注 注意随着 n 的增大，在寻找 $2^{2^n}+1$ 的因子中这个加速方法的效果越来越不明显。比如第 6 个费马数 $F_6 = 2^{2^6}+1 = 18446744073709551617$，由 Pepin 准则知其为合数，在一个台式机上用解释（非编译）语言执行几秒钟即可得出结果。这个数若用试除法则太大了，但是如果用欧拉的加速方法，会在几秒钟内找出因子 274117，尽管我们知道在找出一个因子前需要进行

$$\sqrt{18446744073709551617}/128 \approx 33554432$$

次试除法。用 Pollard 的 Rho 方法也能在几秒内找到分解：

$$F_6 = 274177 \cdot 67280421310721$$

利用基为 2、3、5、7、11、13、17、19 的米勒–罗宾检验只能指出 67280421310721 是一个

可能的素数（不到 1 秒）。但这个数要用除法试验在合理的时间内，如几分钟，验证其素性则显得太大而难以实现。但是，如对这个数应用欧拉的加速方法，只需要用数 $d=1 \bmod 128$（因为 $2^{6+2}=128$）去除，这将需要约

$$\sqrt{67280421310721}/128 \approx 64081$$

次试除法。欧拉加速方法可以在几秒内验证 67280421310721 的素性。

第 7 个费马数

$$F_7 = 340282366920938463463374607431768211457$$

由 Pepin 检验可知其为合数。然而用试除法来寻找一个因子将花费极其大量时间，即使用欧拉加速方法和 Pollard 的 Rho 方法也不能在几分钟内（在相同条件下）找到一个因子：我们需要

$$\sqrt{\sqrt{340282366920938463463374607431768211457}} \approx 4294967296$$

次循环才能找到一个 F_7 平方根附近的因子。

证明（欧拉引理） 如果 $p \mid 2^{2^n}+1$，则 $2^{2^n} = -1 \bmod p$，所以 $2^{2^{n+1}} = 1 \bmod p$。那么我们可断言 2 在 $\mathbf{Z}/p^{\times}$ 中的阶为 2^{n+1}。由费马小定理，$2^{p-1} = 1 \bmod p$，因此 2 的阶整除 $p-1$，即 $2^{n+1} \mid p-1$。这已快要接近我们要证明的结论了，但为了使结论更加明晰，继续如下过程。对 $n \geqslant 2$，也即 $p = 1 \bmod 8$，由二次互反律，$\left(\frac{2}{p}\right)_2 = 1$，也就是说 2 是模 p 的一个平方。由欧拉准则，$2^{(p-1)/2} = 1 \bmod p$。因此，2 在 $\mathbf{Z}/p^{\times}$ 中的阶整除(p−1)/2，由此即得引理的结论。 ♣

备注 如果不用二次互反律，我们将得到一个稍弱一点的结论，即整除 $2^{2^n}+1$ 的 p 具有 $1+k \cdot 2^{n+1}$ 的形式，其中 k 为某个整数。

备注 针对 $2^{2^n}+1$ 的素因子的欧拉条件容易推广（略去二次互反律的应用）为这样的结论：b^n+1 的任何素在子 $p>2$，满足 $p = 1 \bmod 2n$。这又十分类似于费马对 b^n-1 因子的发现，实际上更简单。

备注 利用大的有限域，一个某种程度上更成熟的讨论将导致形成对梅森数 2^n-1 素性的 **Lucas-Lehmer 检验**：定义序列

$$u_0 = 4, u_1 = u_0^2 - 2, u_2 = u_1^2 - 2, \cdots, u_{n-2} = u_{n-3}^2 - 2$$

当 $n>2$ 时，这个数是素数当且仅当 n 为素数，并且 $u_{n-2} = 0 \bmod 2^n - 1$。

当 n 为 3、5、7、13、17、19、31、61、89、107、127、521、607、1279、2203、2281、3217 时，梅森数 2^n-1 为素数。这些数是很小的，可以在一台台式机上很容易地运行 Lucas-Lehmer 检验以验证它们的素数。另一方面，列表上的 n 值对应的梅森数却是很大的，以至于用任何一般的方法都无法证明它们的素性。比如

$$2^{127}-1 = 170141183460469231731687303715884105727$$

有 39 位（十进制），这是 1947 年已知的最大素数。而

$$2^{3217}-1=$$
$$259117086013202627776246767922441530941 8$$
$$188875531254273039749231618740192665863 6$$
$$208620120951680048340655069524173319417 7$$
$$441689509238807017410377709597512042313 0$$
$$666240829163535179523111861548622656045 4$$
$$769112759584877561056875793119101771140 8$$
$$826252153849035830401185072116424747461 8$$
$$230314713983402292880745456779079410372 8$$
$$823582070589235106843388298688861665865 0$$
$$280927692080339605869308790500409503709 8$$
$$759021190183719916209940025689351131365 4$$
$$882973911265679730324198651725011641270 3$$
$$509705427773477972349821676443446683831$$
$$193225400996489940517902416240565190544 8$$
$$369080961606162574304236172186333941585 2$$
$$426431208737266591962061753535748892894 5$$
$$996291951830826218608534009379328394202 6$$
$$186658614250325145077309627423537682293 8$$
$$649407127700846077124211823080804139298 0$$
$$870575047138252645714483793711250320818 2$$
$$612656664908425169945395188778961365024 8$$
$$405739378594599444335231188280123660406 2$$
$$624686092121503499375847822922371443396 2$$
$$885848593821573882123239368704616067736 2$$
$$909315071$$

有 970 位。在 1998 年底已知的最大素数是梅森数 $2^{3021377}-1$。

备注 [Fellows,Koblitz 1992]给出了一个**确定性多项式时间内分解 N 的算法**，条件是假设已知 $N-1$ 的分解。

习题

24.3.01 在 Proth 推论的特殊情况下，利用 Pocklington-Lehmer 准则证明 193 的素性，即将 193 表示为 $193=u\cdot 2^k+1$ 的形式，这里 u 为奇数。

24.3.02 在 Proth 推论的特殊情况下，利用 Pocklington-Lehmer 准则证明 241 的素性。

24.3.03 在 Proth 推论的特殊情况下，利用 Pocklington-Lehmer 准则证明 353 的素性。

24.4 强素数

一个**强素数** p 是这样一个素数，如果它作为一个大整数 n 的因子出现，会使得 n 的分解相当困难。因此，真正的含义依赖于特定因子分解攻击的思想，在这样的攻击中，p 的出现将导致最坏情况的出现。

事实上，“强素数”的概念可能会随时间变化而变化，随着因子分解技术的发展而改变，随着相关的数的尺寸的增加而改变，随着密钥空间的增加而改变，随着计算机运行速度的提高而改变等等（随着算法的改进而改变？）。

准确地讲，一个**强素数**是这样一个素数 p，使得

- $p-1$ 有一个“大”的素因子 r；

- $p+1$有一个“大”的素因子；
- $r-1$有一个“大”的素因子。

第一个条件保证，如果一个整数n有一个因子p，则n可以对抗 Pollard 的$p-1$分解方法，即p不是非常平滑的。第二个条件是对抗对n的类似攻击的需要，对n的一个素因子p，尝试用$p+1$的任何可能的平滑性。第三个条件相对复杂，但实质是一样的。

备注 以前，“$p+1$的一个大素因子s应当本身有一个大的因子”这个条件是强加到“强素数”上的。但现在这个条件不再强加给强素数了，因为我们已经认识到，针对这类防护机制的因式分解攻击已随着数的增大效果越来越差。同样，$r-1$的一个大素因子s也可能被要求$s-1$有这样一个大的素因子。但是针对这种保护的“平滑性”攻击也随着有关数的增大而不再有效。

如下大致描述的算法需要做一些校正以达到上述目标。为了更加明确地说明问题，我们将讨论如何寻找 256 位的强素数。注意在整个过程中，有些参数我们没有指定，但它们的选取仍需要仔细斟酌。

- 选择一个随机的 256 比特的奇整数r；
- 利用米勒–罗宾方法检验$r, r+2, r+4, r+6, r+8, r+10, \cdots$的强伪素性（拟素性）；（当有一个“明显”的小素因子时为避免指数运算，可对每一个数做试除法）
- 设p_1'是找到的（强伪）素数；
- 由素数理论，在找到一个（伪）素数之前，将需要$\frac{1}{2}\log r \approx 89$次尝试。每一个要移位2，因此$p_1 \approx r+2\cdot\frac{1}{2}\log r = r(1+\frac{\log r}{r})$，由此

$$\log_2 p_1 \approx \log_2 r + \log_2(1+\frac{\log r}{r}) \approx \log_2 r = 256$$

  所以我们希望p_1'最可能是一个 256 比特的（伪）素数。
- 然后检验$2p_1'+1$、$4p_1'+1$、$6p_1'+1$、$8p_1'+1$和$10p_1'+1$的强拟素性；
- 设p_1是找到的（伪）素数；
- 由算术级数中素数的狄利克雷定理，作为一种推测，我们设想在找到一个素数前，将需要$\frac{1}{2}\log p_1' \approx 91$次尝试，那么$p_1$将大约有

$$256+\log_2(2\frac{1}{2}\log(p_1')) \approx 256+\log_2(\frac{1}{2}2^{256}) \approx 256+6.5 \approx 263 \text{比特}$$

- 我们用寻找p_1'类似的方法来寻找p_2：选择一个随机的 256 比特的奇数r，并检验$r, r+2, r+4, r+6, r+8, r+10, \cdots$的强伪素性（拟素性），直到找出一个（伪）素数。
- 设p_2是找到的（伪）素数，由素数定理，我们希望在找到一个素数之前，大约进行了$\frac{1}{2}\log r \approx 89$次尝试，$p_2 \approx r+2\cdot\frac{1}{2}\log r = r(1+\frac{\log r}{r})$。由此

$$\log_2 p_2 \approx \log_2 r + \log_2(1+\frac{\log r}{r}) \approx \log_2 r = 256$$

  则p_2最可能是 256 位的素数。
- 现在用欧几里得算法求一个整数t使得$\begin{cases} t=1 \mod p_1 \\ t=-1 \mod 4p_2 \end{cases}$，我们在模数中加入因子 4，

目的是保证将 t 为奇数且 $t=3 \bmod 4$。

- 给定一个解 t，则任何 $t \pm l 4p_1p_2$ 也是解，所以我们可以校正 t，使（不失一般性）$4p_1p_2 < t < 8p_1p_2$。
- 然后用米勒-罗宾方法检验 $t+4p_1p_2$、$t+8p_1p_2$、$t+12p_1p_2$、$t+16p_1p_2$、$t+20p_1p_2$,… 的强伪素性。再一次利用算术级数中素数的狄利克雷定理，我们设想这将需要大约 $\frac{1}{2}\log(t) \approx \frac{1}{2}\log(4p_1p_2) = \frac{1}{2}(2+\log p_1+\log p_2) \approx 180$ 次尝试才能找到一个素数。
- 设 p 是第一个所找到的（伪）素数，故我们希望（再一次利用狄利克雷定理）

$$p \approx t + 180 \cdot 4p_1p_2 \approx 724p_1p_2 \approx 2^{528.5}$$

我们发现产生的素数比我们真正需要的要大一些，因此这个算法需要按下面两种方法进行调整：

- 提前尝试弥补，数的增大是以粗略可估计的量增加的；
- 拒绝仅仅过大的伪素数。这是必要的，因为素数定理和狄利克雷定理的使用仅仅是**试探性的**：定理是正确的，但它们并没真正说明我们的要求是什么。

也就是说这个结论仅仅是一个推测：

- 在 x 附近，大约有 $1/\log x$ 个数是素数；
- 在 x 附近，大约有 $1/\varphi(N)\log x$ 个数是模 N 同余 b 的素数，这里 b 与 N 互素。

24.5 素性证书

利用 Pocklington-Lehmer 准则，证明那些不具有特殊形式的数的素性是很难的，而形式古怪的数往往可能提供可容易验证的**素性证书**。这使得接收方不得不用适量的工作来验证素性，即在多项式时间内。

备注 在下面给出的例子中，我们从头开始来寻找可以提供素性证书的数据。在这个精心设计的例子表明，可以证明一些精选的数的素性，即使在一般方法失效的情况下。

比如，用前面的 Lucas-Pocklington-Lehmer 定理来证明 N 为素数，我们只要分解 $N-1=K \cdot U$ 就足够了。

- 这里 K 的分解是完全已知的；
- $K > \sqrt{N}$；
- 对每个整除 K 的素数 q，寻找 b_q，使得 $b_q{}^{N-1} = 1 \bmod N$，但是 N 与 $b_q{}^{(N-1)/q} - 1$ 互素，即 $gcd(b_q{}^{(N-1)/q} - 1, N) = 1$。

  否则，再利用第二种方法，仍然只要分解 $N-1=K \cdot U$。
- 这里 K 的分解是完全已知的；
- 未分解部分 U 没有小于等于 B 的素因子，B 为某个界限；
- $B \cdot K > \sqrt{N}$；
- 对每个整除 K 的素数 q，寻找 b_q，使得 $b_q{}^{N-1} = 1 \bmod N$，但是 N 与 $b_q{}^{(N-1)/q} - 1$ 互素，即 $gcd(b_q{}^{(N-1)/q} - 1, N) = 1$；
- 寻找 b_0，使 $b_0{}^{N-1} = 1 \bmod N$ 但 $gcd(b_0^K - 1, N) = 1$。

在上述两种方法中，K 的分解、所找到的一系列 b 以及素数 q 合在一起构成了 N 的**素性证书**。

当然，这个证书必须包括 K 的分解中出现的小素数 q 的素性证书，它还需要在这些证书中出现的素数的素性证书，等等。

比如，$N=1000000033$，运用基为 2、3、5、7、11 的米勒-罗宾方法。我们发现这是一个可能的素数，容易去掉 $N-1$ 的几个小因子：

$$N-1=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 3\cdot 127\cdot 82021$$

这里我们可以很容易地验证 2、3 和 127 的素性。假定我们用试除法后，发现 $U=82021$ 没有小于等于 127 的素因子。采用上面的记号，即 $B=127$，取 $K=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 3\cdot 127=12192$。条件 $K\cdot B\approx 1548384>\sqrt{N}\approx 31623$ 满足。

现在我们对整除 $N-1$ 的素数寻找 b，当然对剩余部分 U 也要如此。希望 N 是素数，条件 $b^{N-1}=1 \bmod N$ 应当是容易满足的。比较困难的是条件 $gcd(b^{(N-1)/q}-1,N)=1$。首先，$q=2$ 整除 $N-1$，

$$2^{(N-1)/2}=1 \bmod N$$

$$3^{(N-1)/2}=1 \bmod N$$

即 2 和 3 都不满足条件。而 $5^{(N-1)/2}=-1 \bmod N$，因此 $b_2=5$ 满足条件（因为 N 为奇数，如果 $A=-1 \bmod N$，则 $A-1=-2 \bmod B$ 且 A 必与 B 互素）。对于 $N-1$ 的素因子 $q=3$，我们还是试一下 $b=5$：

$$5^{(N-1)/3}=566663896 \bmod N$$

且由欧几里得算法表明 $gcd(5^{(N-1)/3}-1,N)=1$，因此 $b_3=5$。对于 $N-1$ 的素因子 $q=127$，

$$5^{(N-1)/127}=915796555 \bmod N$$

经验证 $gcd(5^{(N-1)/127}-1,N)=1$，因此 $b_{127}=5$。

最后，因为我们用的是经校正的第二种方法，不需要找 b_0，使 $b_0{}^{N-1}=1 \bmod N$ 但 $gcd(b_0^K-1,N)=1$。再一次用 $b_0=5$ 试验，因为前面讨论已知第一个条件是满足的，即 $5^{N-1}=1 \bmod N$。而由欧几里得算法表明 $gcd(5^K-1,N)=1$，因此数据 K = 2·2·2·2·2·3·127 = 12192 以及一系列 $b_2=b_3=b_{127}=b_0=5$ 就是 $N=1000000033$ 的**素性证书**。

备注 一旦给出这些信息，则所需做的所有验证可以非常有效地完成（即在多项式时间内）。也就是说，素性证书把一些工作留给了观察者，但余下的工作必须是多项式时间的工作，而非像大数的试除法那样难以承担的工作。

备注 在这个例子中，实际上不需要大量的工作就能找到组成证书的那些数据。但也存在那样的情况，需要大量的工作才能找到证书数据。

备注 注意在本例中，我们并没有关注 $U=82021$ 为素数这个事实，仅关心它没有小于等于 127 的素因子。这样节约了 75 次试除法。

作为一个更加真实的例子，我们来考虑

$$N=3^{53}-2^{53}=193832456586728206420557 31$$

运用米勒-罗宾检验表明这是一个可能素数。我们仍用 Pocklington 定理来给出它的素性证书。尝试完全分解 $N-1$ 可能是不合理的，因此我们只搜寻 10000 以内的素数因子：

$$N-1=2\cdot 3\cdot 5\cdot 7\cdot 53\cdot 263\cdot 6621792797417598667$$

运用基为 2 的米勒-罗宾方法，我们在上式的最后一个数

$$t=6621792797417598667$$

确定地为一个合数（尽管它没有 10000 以内的素因子）。由于 t 还不算太大，可以有效应用

Pollard 的 Rho 方法：我们容易发现（仅在 592 次循环后）$t = 906043 \cdot 7308475201969$，容易验证（用机器）906043 为素数。米勒-罗宾的方法表明 t 的分解式中那个大因子 $u = 7308475201969$ 为一个可能素数。我们用 Pocklington 定理来验证这一点：分解 $u-1$ 为

$$u-1 = 2 \cdot 2 \cdot 2 \cdot 3 \cdot 83 \cdot 17539 \cdot 104593$$

而且这些因子容易验证是素数。为了证明 $N-1$ 的因子 u 为素数，我们需要验证它满足 Pocklington-Lehmer 的条件，但 2、3、5、7 均不能充当 b_2。第一个符合条件的 b_q 为 $b_2 = 11$，因为 $11^{u-1} = 1 \bmod u$，且 $gcd\,(11^{(u-1)/2} - 1, u) = 1$。进一步，$gcd\,(3^{(u-1)/3} - 1, u) = 1$，并且 $3^{u-1} = 1 \bmod u$，所以 $b_3 = 3$。类似地有

$$gcd(3^{(u-1)/5} - 1, u) = 1$$
$$gcd(3^{(u-1)/7} - 1, u) = 1$$
$$gcd(3^{(u-1)/53} - 1, u) = 1$$
$$gcd(3^{(u-1)/263} - 1, u) = 1$$

因此我们已验证 u 为素数。现在返回来看 $N-1$，

$$N-1 = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 53 \cdot 263 \cdot 906043 \cdot 7308475201969$$

而此时我们已经知道这些因子均为素数。我们对 N 来运用 Pocklington-Lehmer 准则：$2^{N-1} = 1 \bmod N$ 且 $gcd(2^{(N-1)/2} - 1, N) = 1$，所以 $b_2 = 2$ 对 N 有效，但它不能作为 b_3，3 也是如此。但我们发现 5 满足诸多条件：$5^{N-1} = 1 \bmod N$ 且

$$gcd(5^{(N-1)/3} - 1, N) = 1$$
$$gcd(5^{(N-1)/5} - 1, N) = 1$$
$$gcd(5^{(N-1)/7} - 1, N) = 1$$
$$gcd(5^{(N-1)/53} - 1, N) = 1$$
$$gcd(5^{(N-1)/263} - 1, N) = 1$$
$$gcd(5^{(N-1)/906043} - 1, N) = 1$$
$$gcd(5^{(N-1)/7308475201969} - 1, N) = 1$$

因此，$b_3 = b_5 = b_7 = b_{53} = b_{263} = b_{906043} = b_{7308475201969} = 5$（按照前面的记号，最后一个应当记为 b_0），故这些数据可作为 N 的素性证书，证明 N 为一个素数。

最后我们来给出一个例子，普通的方法对它无能为力：我们将寻找一个大数 N，使得 $N-1$ 有几个大素数因子，这将保证如果事先不知道这些因子的情况下，分解 $N-1$ 相当困难。我们将使用前面的素数 N，而且还不止需要一个大素数。同选择前面的 N 一样，随机选择一个数，令 $M = 1938324565867283064205576 7$，由米勒-罗宾方法知道这是一个可能素数。在给出后面的例子之前，必须证明它是素数，用 10000 以内的素数的试除法我们发现：

$$M-1 = 2 \cdot 11 \cdot 41 \cdot 47 \cdot 307 \cdot 653 \cdot 2280712525657409$$

令 $B = 10000$，将小因子相乘并乘以 B 后，该值大于 M 的平方根。而且通过试除法我们知道 $u = 2280712525657409$ 没有小于 B 的素因子。我们发现 $2^{M-1} = 1 \bmod M$，并且 $gcd(2^{(M-1)/2} - 1, M) = 1$，因此 $b_2 = 2$。进一步，如果运气好的话，可以得到：

$$gcd(2^{(M-1)/11} - 1, M) = 1$$
$$gcd(2^{(M-1)/41} - 1, M) = 1$$
$$gcd(2^{(M-1)/47} - 1, M) = 1$$
$$gcd(2^{(M-1)/307} - 1, M) = 1$$

$$gcd(2^{(M-1)/653}-1, M)=1$$
$$gcd(2^{(M-1)/2280712525657409}-1, M)=1$$

这就证明 M 为一个素数。

现在我们来构造一个更大的素数。M 和 N 如前，我们来考虑**算术序列**：

$$n=2MN+1,\quad 4MN+1,\quad 6MN+1,\quad 8MN+1,\quad \cdots$$

直到一个数 n 满足米勒-罗宾条件，然后我们利用 Pocklington-Lehmer 方法证明 n 的素性。这里的技巧就是利用 n 的特殊形式，因为我们事先知道 $n-1$ 有因子 M 和 N，而且希望其他的因子也不至于太可怕。如果我们事先不知道 M 和 N 是因子，则要找出它们是相当困难的。我们发现

$n=2\cdot 78\cdot M\cdot N+1=586107931132555950104046859251895127208517525363056 13$ 是一个可能素数。由于 $n-1=2\cdot 78\cdot M\cdot N$，我们可以很容易将 $n-1$ 分解为素数的乘积：

$$n-1=2\cdot 2\cdot 3\cdot 13\cdot M\cdot N$$

而且我们很幸运地发现 2 可作为 b_2、b_3、b_{13}、b_M 和 b_N。因此，这个数是素数。

备注 在这个例子中我们还发现 2 是模 n 的一个本原根。

备注 上述各步骤的计算是在一台主频为 200MHz 的机器上用解释语言 Python 执行的，并且所有指出的计算几乎在不易觉察到的时间就完成了，由于是用解释语言执行的，要比在相同机器条件下用完全编译的语言执行慢 10 倍左右。计算能力越强，计算工作越容易完成。另一方面，即使有一台很好的机器，单纯用穷尽试验方法来证明上述例子中 n 的素性是不可取的：因为这个 n 为 54 位的十进制数，故至少需要 $10^{27}/(27\ln 10)\approx 10^{25}$ 次试除法（这里除以 $27\ln 10$ 是用素数定理所做的推测性估计）。即使每秒可做一万亿次试除法，要做完上例中 n 的素性证明所需要的除法，需要 1 亿年。

另外，如果我们不知道大数 M 和 N 的信息而利用 Pocklington-Lehmer 方法，仍假设每秒可进行一万亿次试除法，要用除法试验分解 $MN\approx 10^{52}$ 也需要花费 1000 万年。

即使用合理的古典分解算法，比如 Pollard 的 Rho 方法，要分解上面的 $n-1$ 也需要很长时间。

第25章 现代因式分解攻击

二次筛法分解算法是最基本的“现代”因式分解方法，它的适当优化甚至可以与一些成熟的方法如椭圆曲线分解或数域筛法不相上下。通常该算法有一个概率元素，但一旦找到了大数的一个真因子，就可以很确定地验证这个因子的确是那个数的真正的因子。

二次筛法是对随机平方分解方法的思想进行提炼的结果，而**随机平方因子分解**方法则源自于费马的方法。按照现在的标准，二次筛法可以比得上**椭圆曲线筛法**（并且快一些），对于超过115位或120位的十进制整数，二次筛法要比**数域筛法**慢。对这些算法的描述还需要做一些必要的准备工作。

在下面的第一节，首先介绍一点线性代数的知识——**高斯消元法**，它在二次筛法最原始形式的运行中也是非常重要的。

25.1 高斯消元法

高斯消元法是寻找任意可计算域上向量空间中向量的**线性相关关系**的一个有效算法。我们现在更感兴趣的是有两个元素的有限域$\mathbf{F}_2=\mathbf{Z}/2$上$n$元组构成的向量。

备注 通常这个算法的讨论会涉及到使用浮点实数运算的情况，在这种情况下，精度的损失问题尤为重要。但在我们现在的讨论中，所计算的“数”取自于任意有限域，我们使用**无限精度**，所以不必担心舍入误差。这样就可避免在浮点情况下过多地考虑这类问题。

我们需要解决的问题如下所述：令

$$\begin{aligned}v_1&=(v_{11},v_{12},\cdots,v_{1,n})\\v_2&=(v_{21},v_{22},\cdots,v_{2,n})\\v_3&=(v_{31},v_{32},\cdots,v_{3,n})\\&\cdots\\v_m&=(v_{m1},v_{m2},\cdots,v_{m,n})\end{aligned}$$

为域k上元素的n元组，域k可以是有理数域$\mathbf{Q}$、实数域$\mathbf{R}$、有限域$\mathbf{F}_p=\mathbf{Z}/p$（$p$为素数）或其他域。尽管我们**精确地**知道数，或者说数有**无限的精度**，我们仍然要对它们进行运算。这个假设一般是不满足的，比如数值是来源于测量结果的实数。但在这里的应用中，我们假设条件是成立的。问题就是寻找一组k中不全为0的元素$c_1,\cdots,c_m$，使得

$$c_1v_1+\cdots+c_mv_m=0$$

这里指出的乘法为标量乘法$c(x_1,x_2,\cdots,x_m)=(cx_1,cx_2,\cdots,cx_m)$，并且右边的0为**零向量**$0=\underbrace{(0,\cdots,0)}_{n}$。形如

$$c_1v_1+\cdots+c_mv_m$$

的表达式称为**向量**v_i以c_i为**系数**的**线性组合**，并且任何形如$c_1v_1+\cdots+c_mv_m=0$（系数不全为0）的关系称为一个**线性相关关系**。向量的**维数**即为n元组中的n。

备注 在基础线性代数的系统化发展中，首先被证明的一个结论就是：如果向量的个数大于维数，则存在一个线性相关关系。我们将不直接使用这一事实，而是希望验证我们所描述的算法果真如同它所声明的那样有效。

首先，我们由向量的分量构造一个 $m\times n$ 矩阵，令

$$M=\begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ v_{31} & v_{32} & \cdots & v_{3n} \\ & \cdots & & \\ v_{m1} & v_{m2} & \cdots & v_{mn} \end{pmatrix}$$

然后通过在该矩阵右边附加一个 $m\times m$ 单位矩阵的方法构造一个更大的矩阵 $\widetilde{M}$，我们来看一下 $\widetilde{M}$：

$$\widetilde{M}=\begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1n} & 1 & 0 & 0 & \cdots & 0 \\ v_{21} & v_{22} & \cdots & v_{2n} & 0 & 1 & 0 & \cdots & 0 \\ v_{31} & v_{32} & \cdots & v_{3n} & 0 & 0 & 1 & \cdots & 0 \\ & \cdots & & & & \cdots & & & \\ v_{m1} & v_{m2} & \cdots & v_{mn} & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

那个单位矩阵将保留我们所做变换的轨迹。

这里合法的变换称为**初等行变换**，它们是：

- 互换两行的位置；
- 用非零常数乘以某一行；
- 从一行中减去另一行的倍数。

利用初等行变换对 $\widetilde{M}$ 施行行变换的过程如下。从最左边一列开始，如果其最顶端元素为 0，但该列中有一些元素不为 0，那么互换行的位置使最顶端元素为非零。用（新的）第一行的最左边元素除以整行，使最左边元素为 1。设在第 i 行的最左边元素为 a_{i1}，则对 $i>1$，从其余各行中减去第一行的 a_{i1} 倍，这就使得第一列中除了最顶端元素为 1 外，其余全为 0。（如果最左边或任意其他列全为 0，则忽略它。）

接下来再看第二列，如果必要，可以互换第二行与其下面另一行的位置，目的就是经调整后第二行的第二个元素非零（第一行下面的各行中，第一个元素已全为 0）。用第二个元素除以该行，使得第二行以 0、1 开始。设在第二列中，从顶端开始的第 i 个元素为 a_{i2}，然后从所有其他各行中减去第二行的 a_{i2} 倍（包括顶行）。

对第三、第四直到第 m 列进行类似的过程，直到前 m 列中其余元素全为 0。令 A 为由这一变换过程获得的矩阵右边的新的 $m\times m$ 矩阵，并令 $w_1,\cdots,w_m$ 为新矩阵的 n 元组的 m 个行（长为 n 的行向量），则有

$$A\begin{pmatrix} v_1 \\ v_2 \\ \cdots \\ v_m \end{pmatrix}=\begin{pmatrix} w_1 \\ w_2 \\ \cdots \\ w_m \end{pmatrix}$$

如果 $m>n$，则至少有最后 $m-n$ 个 w_i 为零向量。比如 w_m 是长为 n 的零向量。即有

$a_{m1}v_1 + a_{m2}v_2 + a_{m3}v_3 + \cdots + a_{mn}v_m = (0,\cdots,0)$，对每个指标 i，至少有一个 a_{ij} 为非零，这一点是相当重要的。

换句话说，我们找到了向量 v_i 的一个线性组合为零向量（但线性组合中的系数不全为 0）。

25.2 随机平方分解

二次筛法发展的第一阶段就是**随机平方**因式分解方法。这一方法的思想源自于 350 年前费马所用的方法。我们这里给出它的描述，但它不是一个实用的算法，因为我们没有一个确定的程序来执行某些必要的步骤。但是，这里的讨论可以解释它的另一种形式。

这里的一个主要思想就是在寻找整数 n 的因子的过程中，如果 $x^2 = y^2 \bmod n$，但 $x \neq \pm y \bmod n$，则 $gcd(x-y,n)$ 即为 n 的一个真因子。实际上，这里的假设就是 n 整除 $(x-y)(x+y)$，但 n 不能整除 $x-y$ 也不能整除 $x+y$。也就是说 n 整除两个因子的乘积，但不能整除任何一个因子。因此，$gcd(x-y,n) \neq n$ 且 $gcd(x+y,n) \neq n$。然而这两个最大公因子中没有一个可能是1和 n。因此，问题就转化为对给定的 n，如何系统有效地寻找这样的 x 和 y。

我们来看一下同余式 $x^2 = y^2 \bmod n$，$x \neq \pm y \bmod n$ 成立的机会很大。设 k 是能整除 n 的不同奇素数的个数，并假设 n 为奇数（因为我们可以很容易地检验并去掉因子 2）。这样我们声称：对一个平方 $b = x^2 \bmod n$，存在 2^k 个不同的模 n 的平方根 b。实际上，令

$$n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$$

则单个同余式 $x^2 = b \bmod n$ 就等价于（由孙子定理）如下 k 个类似同余式的方程组：

$$\begin{cases} x^2 = b \bmod p_1^{e_1} \\ \cdots \\ x^2 = b \bmod p_k^{e_k} \end{cases}$$

令 $\pm a$ 是 b 模 n 的平方根，则同余方程组又等价于：

$$\begin{cases} x = \pm a \bmod p_1^{e_1} \\ \cdots \\ x = \pm a \bmod p_k^{e_k} \end{cases}$$

这里很重要的一点就是 $\pm$ 可以互相独立地选择，即因为对 $\pm$ 分别有 k 个不同的选择，故有 2^k 个不同的同余方程组，而由孙子定理可知每一个都有一个模 n 的解。在 2^k 个解中仅有 2 个是显然的解 $\pm a$，因此两个解 x 和 y 恰好满足同余式 $x = \pm y \bmod n$ 的概率仅为 $2/2^k$。

因此，问题转化为寻找足够的数对 x 和 y 满足 $x^2 = y^2 \bmod n$，以使某对 x_0, y_0 满足 ${x_0}^2 = {y_0}^2 \bmod n$ 但 $x_0 \neq \pm y_0 \bmod n$ 的概率足够高。这一思想下面还要展开描述。

25.3 Dixon 算法

二次筛法最简单的应用形式就是 **Dixon 算法**，我们给出该算法的简要描述。这里还需要引入**因子基**的概念，它与一些问题有关。

因子基就是前 t 个素数的集合 $S = \{p_1, p_2, \cdots, p_t\}$（即 $p_1 = 2$，$p_2 = 3$，等等）（如何选取适当的 t 将在下面讨论）。我们选取整数 $a_1, a_2, \cdots$ 并用模 n 约简，令 $b_i = a_i^2 \% n$。我们要求每个 b_i 的出现是关于因子基 S **平滑**的，即要求 b_i 是 p_t **平滑**的。然后找出一个 b_i 的子集，其元素之积为 **Z** 中的完全平方。这一步骤按如下方式进行，将 b_i 分解为

$$b_i = \prod_{j=1}^{t} p_j^{e_{ij}}$$

假设有$t+1$个满足$b_i = a_i^2 \%n$的a_i，且b_i的因子全在S中。令v_i为$\mathbf{F}_2^{t+1}$中的**向量**，它的各个分量由b_i的素因子分解中指数模2后构成：

$$v_i = (e_{i1}\%2, e_{i2}\%2, e_{i3}\%2, \cdots, e_{it}\%2)$$

因为在t维空间中有$t+1$个这样的向量，它们在$\mathbf{F}_2$上是**线性相关的**。即存在$c_1, \cdots, c_t$，使

$$c_1 v_1 + \cdots + c_t v_t = (0, \cdots, 0) \in \mathbf{F}_2^{t+1}$$

利用高斯消元法来寻找这样的关系，则有

$$\prod_{i=1}^{t+1} b_i^{c_i} = \prod_{i=1}^{t+1} \left(\prod_{j=1}^{t} p_j^{e_{ij}} \right)^{c_i} = \prod_{j=1}^{t} \prod_{i=1}^{t+1} p_j^{c_i e_{ij}} = \prod_{j=1}^{t} p_j^{\sum c_i e_{ij}}$$

它有偶指数，所以是一个整数的平方。

这里，取$x = \prod_{i=1}^{t-1} a_i^{c_i}$，$y = \prod_{j=1}^{t} p_i^{(\sum c_i e_{ij})/2}$。我们的讨论表明$x^2 = y^2 \bmod n$。因此，假设$n$真的是一个合数（非素数方幂），并假设所有的选择都是充分“随机”，那么我们就有50%的可能得到$1 < gcd(x-y, n) < n$，即获得n的一个真因子。

如果很不幸地得到$x = \pm y \bmod n$，则得不到n的真因子。这种情况下，重新计算一个或多个新的$b_i = a_i^2 \%n$，然后重复上述高斯消元法。

备注 根据向量空间$\mathbf{F}_2^t$上更加详细的讨论，很明显在向量v_i中存在几个线性相关关系，那么在这些关系中至少有一个会产生$x \neq \pm y \bmod n$的可能性就会很大。

问题被转化为两件事情：一是适当选取因子基$S = \{p_1, p_2, \cdots, p_t\}$中素数的个数$t$；二是有效地选择$a_i$，使$b_i = a_i^2 \%n$的素因子分解中的素因子属于$S$。**Dixon算法**中$a_i$是随机选取的，并且用$S$中的素数做**试除法**，检验$b_i = a_i^2 \%n$是否为$p_t$平滑的。这就需要做$t$阶次试除法。如果$t$可能小，这种检验还不是太无效。如果$b_i = a_i^2 \%n$不是$p_t$平滑的，只要拒绝这个$a_i$，重新选择一个。这就是Dixon算法的基本形式，在我们给出例子之前对它做一点改进。

备注 对于一个较小的界B，满足$2 \leqslant b \leqslant B$的数$b$中有一部分是$p_t$平滑的，这从直觉上是显然的，而且也可以证明。但是，一方面，如果B太小的话，将需要花费很长时间去寻找满足$b_i = a_i^2 \%n$是B平滑的随机$a_i s$。因此，我们需要这样一种机制，它能以某种方式选择a_i，使$b_i = a_i^2 \%n$相对小。最有效的机制就是目前已知的**二次筛法**，稍后给出。

备注 **避免素数方幂**：如果碰巧n是某个（奇）素数的方幂，这个方法将完全失效。因为利用亨泽尔引理可证明，最多存在两个模n的平方根。因此，不仅要用米勒-罗宾检验保证n是一个合数，而且要进行额外的检验以保证n不是一个素数的方幂。当然应当有特殊的方法来分解已知是一个素数方幂的大整数。

备注 对给定的n，t最优的选择是大约$t = e^{\frac{1}{2}\sqrt{\ln n \ln \ln n}}$，这来自于有关$\sqrt{n}$附近整数平滑性的更加详细的信息。由$t$的估计值，以及将**筛法**代替试除法作为二次筛法算法中平滑性检验的方法，给出了一个推测性的运行时间估计为

$$O(e^{(1+o(1))\sqrt{\ln n \ln \ln n}})$$

更进一步，这个运行时间估计独立于素数因子的大小。

25.4 非筛的二次筛法

二次筛法的第一个版本实际上并没有像其名字那样的**筛**的含义，稍后我们会给出解释。但它利用了 a_i 选择的最优化方法，使这种形式可用。与 Dixon 一样，希望在数对 (a_i,b_i) 的选取上更聪明些，并且用两种方法对因子基 $S=\{2,3,\cdots,p_t\}$ 做了改进。

假定我们试图分解整数 n，并选取了一个**因子基** $S=\{2,3,5,7,11,\cdots,p_t\}$，选取的 a_i 将保证它刚好大于 n 的平方根，那么对于大于 $\sqrt{n}$ 的 a，取 $b=a^2\%n$。同在任何随机平方分解方法中一样，如果 b 是 S **平滑**的，我们**接受**数对 (a,b)，否则就抛弃它。

备注 如果 a 充分接近 n 的平方根，则相应的 b 可以用特殊的方法计算。令 $m=\text{floor}(\sqrt{n})$，即不超过 $\sqrt{n}$ 的最大整数，$m:\leqslant\sqrt{n}$，则

$$b=((a_i-m)+m)^2-n=(a_i-m)^2+2\cdot(a_i-m)m+m^2-n$$

且 $b_i\%n=(a_i-m)^2+2\cdot(a_i-m)m+m^2-n$，$a_i-m$ 与 m 相比足够小。

备注 如果用特殊方法计算 b 获得了较好的效率，比如使 b 可能是负的，则 -1 应当包含在因子基 S 中。

例子 作为一个分解较小整数的例子，我们用这一方法来分解 $n=143$，因子基则为 $B=\{2,3,5\}$。我们应当以正好大于 n 的平方根的整数开始，所以取 $a_1=12$。然后计算 $b_1=a_1^2\%143=1$，我们的运气还不错：这个值本身就是 B 平滑的且是一个平方。因此，取 $x=a_1=12$，$y=1$（b_1 的正平方根），有关系式 $12^2=1^2 \bmod 143$。计算 $gcd(12-1,\ 143)$ 和 $gcd(12+1,\ 143)$，前面一个最大公因子为 11，后面一个为 13。所以我们就找到了 143 的真因子。

例子 作为另一个分解较小整数的例子，我们来分解 $n=1739=37\cdot47$，因子基选择为 $B=\{2,3,5\}$。同样，我们以正好大于 n 的平方根的整数即 42 开始。$b_1=a_1^2\%1739=25$ 是 B 平滑的且是一个平方。因此，计算 $gcd(42-5,\ 1739)=37$ 且 $gcd(42+5,\ 1739)=47$，所以就找到了 1739 的真因子。

例子 作为一个选择因子基问题的例子，我们来分解 $n=3071=37\cdot83$，因子基为 $B=\{2,3,5\}$。我们以正好大于 $\sqrt{3071}$ 的整数 56 开始，但第一个得到的 B 平滑的平方是 $a_1=96$，且 $b_1=a_1^2\%3071=3$。我们验证接下来一个数不是 B 平滑的，直到 $a_2=125$，且 $b_2=a_2^2\%3071=270$。接下来是 $a_3=157$，且 $b_3=a_3^2\%3071=81$，恰好这是一个平方，那么我们就可以终止算法，并计算 $gcd(157-9,\ 3071)=37$ 和 $gcd(157+9,\ 3071)=83$。这里的问题是我们大约计算了 100 个平方与模 3071 约简，才找到了因子。这比试除法还是要快一些。

备注 因为 a 的值与 n 的平方根相比较大，我们可以修改算法以约简模 n。

例子 用稍大一点的因子基来分解 $n=3071=37\cdot83$，$B=\{2,3,5,7\}$。同前面一样，我们用 $a=56$ 开始，第一个 B 平滑的情况是 $a_1=96$，它比初始值 56 大得多，所以我们不能仅计算 a^2-55，而还应该计算 3071 约简。实际上，$a_2=97$ 就给出了一个完全平方

$$b_2=a_2^2\%3071=196=14^2$$

因此我们找到了真因子 $gcd(97-14,\ 3071)=83$ 和 $gcd(97+14,\ 3071)=37$。这里的步骤也太多了。

例子 我们再来用更大一点的因子基 $B=\{2,3,5,7,11\}$ 来分解 $n=3071=37\cdot83$。仍然从

$a=56$ 开始，第一个 B 平滑的情况是 $a_1=79$，这个值已经比初始值大了不少，并且预示不妙，因为我们正在花费比试除法所需时间还要多的时间。

例子 分解 $n=8383=83\cdot101$，我们从 a=92 开始，由于 $92^2\%8383=81$ 已经是一个平方，所以我们就已经找到了因子 $gcd(92-9,\ 8383)=83$ 和 $gcd(92+9,\ 8383)=101$。

例子 用因子基 $B=\{2,3,5,7,11\}$ 来分解 $n=66887=211\cdot317$。我们以 $a=259$ 开始，直到 $a_1=368$ 才发现平滑的情况，这个值已经比66887的平方根大了不少，我们不能用更简单的形式了，不过我们可以计算：

$$b_1=a_1^{\ 2}\%66887=1650=2\cdot3\cdot5^2\cdot7^0\cdot11$$

下一个平滑的情况为 $a_2=382$，且

$$b_2=a_2^{\ 2}\%66887=12150=2\cdot3^5\cdot5^2\cdot7^0\cdot11^0$$

再下来的一个平滑的情况为 $a_3=697$，且

$$b_3=a_3^{\ 2}\%66887=17600=2^6\cdot3^0\cdot5^2\cdot7^0\cdot11$$

这里有一个捷径，因为我们发现相应的指数向量 (1,1,2,0,1)、(1,5,2,0,0) 与 (6,0,2,0,1) 的和为 (8,6,6,0,2)，每个分量均为偶数。因此对每个分量除以 2 得到 (4,3,3,0,1)，取

$$y=2^4\cdot3^3\cdot5^3\cdot7^0\cdot11^1\%66887=58904$$

再取 x 为 $a_1\cdot a_2\cdot a_3\%n$，即

$$x=\sqrt{1650\cdot12150\cdot17600}\%66887=58904$$

因为 $x=y$，我们自叹运气不佳。继续执行算法，我们会发现更多 B 平滑的值 $a^2\%n$，但我们花费的时间已经远远超过试除法所需要的时间。

例子 取因子基为 $B=\{2,3,5,7,11\}$，我们来分解 $n=2043221=1013\cdot2017$。我们找到如下一些平滑的结果：

$$1439^2\%2043221=27500=2^2\cdot3^0\cdot5^4\cdot7^0\cdot11^1$$
$$2878^2\%2043221=110000=2^4\cdot3^0\cdot5^4\cdot7^0\cdot11^1$$
$$3197^2\%2043221=4704=2^5\cdot3^1\cdot5^0\cdot7^2\cdot11^0$$
$$3199^2\%2043221=17496=2^3\cdot3^7\cdot5^0\cdot7^0\cdot11^0$$
$$3253^2\%2043221=365904=2^4\cdot3^3\cdot5^0\cdot7^1\cdot11^2$$

同样，我们的选择排除了利用更简单方法的可能。注意到前两个指数向量的和向量的每个分量均为偶数，但同前一个例子一样，不幸的是 $x=y$。我们还发现第三和第四个指数向量的和亦为偶数分量的向量：即

$$(5,1,0,2,0)+(3,7,0,0,0)=(8,8,0,2,0)$$

所以用向量中指数的一半，我们取

$$y=2^4\cdot3^4\cdot5^0\cdot7^1\cdot11^0\%2043221=9072$$

再取 x 为对应的 a_3 和 a_4 的乘积：

$$x=(3197\cdot3199)\%2043221=11098$$

然后我们计算最大公因子：

$$gcd(x-y,n)=gcd(11098-9072,\ 2043221)=1013$$
$$gcd(x+y,n)=gcd(11098+9072,\ 2043221)=2017$$

这还是比试除法要慢。因为这个 n 较小，且我们选取的因子基也较小。

另外注意到，如果我们能够保持 a 非常接近 n 的平方根，则对一个可以整除某个 b_i 的素

数 p，$b_i^2 = n \bmod p$。也就是说，这种情况仅对 p 出现，使 n 是模 p 的平方根。通过二次互反律，在任意给定范围内约有半数的素数。因此，对固定的 n，因子基仅需包含满足 $\left(\frac{n}{p}\right)_2 = 1$ 的素数 p。在分解较大的数时，这种方法可以极大地减少计算量，但在整数较小的例子中我们无法利用它的优势。

例子 分解 $n = 20000900009 = 100003 \cdot 200003$，我们希望利用接近 n 的平方根的 a，这样就可以在一个因子基中省略素数 p，如果 n 是模 p 的非平方。在本例中，$n\%3 = 2$，是模 3 的非平方，所以可经略去 3。$n\%7 = 6$，它不是模 7 的平方；$n\%11 = 2$，也不是模 11 的平方；$n\%23 = 5$，也不是模 23 的平方。因此，如果我们已经准备用因子基

$$B = \{2,3,5,7,11,13,17,19,23,29\}$$

那么此时可以将因子基简化为 $B = \{2,5,13,17,19,29\}$，实际上，对 $a = 141427$、145023、150003、167565 和 178547，我们得到了平滑的 $b = a^2\%n$，而且

$$150003^2\%2000090009 = 2500000000 = 50000^2$$

已经是一个平方，那么只需要计算最大公因子就行了：

$$gcd(150003 - 50000,\ 20000900009) = 100003$$

$$gcd(150003 + 50000,\ 20000900009) = 200003$$

这就得到了两个真因子。

25.5 二次筛法

二次筛法的这个精炼形式是名符其实的**筛法**，主要体现在对 a_i 的选择上。当被分解的数很大时这是很有价值的。

所有的假设同上一节。首先我们必须解释促使筛法产生的基本机制，筛法本身稍后描述。注意到，如果 p 是因子基 S 中的奇素数，并且对某个 x，p 整除 $q(x)$，则对每个整数 l，p 也整除 $q(x + lp)$。因此，如果让 l 变化，我们可以由二次方程 $q(x) = 0 \bmod p$ 的一个或两个不同的解 x 得到其他解的两个序列。

筛法描述如下。对一个大数 M，我们构造一个以 x 为下标的数组 A，$-M \leqslant x \leqslant M$。数组的第 x 个值初始化为 $\text{floor}(\lg|q(x)|)$，这里的 lg 为以 2 为底的对数，$\text{floor}(t)$ 是不超过 t 的最大整数。令 x_1, x_2 是方程 $q(x) = 0 \bmod p$ 的两个解，p 是因子基 S 中的奇素数。

现在我们来做**筛法**：对 $x = x_1 \bmod p$ 或 $x = x_2 \bmod p$ 的数组 A 的那些值 $A[x]$，从这些 $A[x]$ 中减去 $\text{floor}(\lg p)$。对因子基 S 中的每个奇素数，重复做这一过程。做完这一过程后，数组中值接近 0 的那些 $A[x]$ 最可能是 p_i 平滑的。这可以用试除法分解 $q(x)$ 来验证。

备注 这里必须考虑舍入误差，但与椭圆曲线算法的成熟应用相比，二次筛法不需要太高的精度。

25.6 其他改进

对前面所述的方法做进一步的改进和优化是可能的，但要解释这样做的理由却是困难的。

备注 我们来看一下对给定的整数 n，t 的最优选择大约为

$$t = O(e^{\frac{1}{2}\sqrt{\ln n}\sqrt{\ln\ln n}})$$

这个估计来自于有关$\sqrt{n}$附近整数平滑性的详细信息。在二次筛法算法中，用**筛法**代替试除法来检验平滑性，这给出了运行时间的估计：

$$L[\frac{1}{2},1]=O(e^{(1+o(1))\sqrt{\ln n}\sqrt{\ln\ln n}})$$

并且这个时间估计独立于素因子的大小。

备注 **多重二次多项式筛法**使用了几个二次多项式，而不是上述方法中的一个多项式。这降低了筛选的时间间隔。渐近的运行时间估计仍为

$$L[\frac{1}{2},1]=O(e^{(1+o(1))\sqrt{\ln n}\sqrt{\ln\ln n}})$$

这似乎是许多实现中所用的形式。

备注 二次筛法特别适合在多重形式中做**并行处理**：每个处理器使用一个不同的二次多项式，并且只将合适的数对a_i和b_i报告给中央处理器。在足够多的a_i和b_i找到后，中央处理器可以求解x，y。

第26章 有限域

尽管我们已经习惯于把有理数域、实数域和复数域当作“自然的”数的集合，但非常重要的一点是我们应认识到还有许多其他重要的域。这也许是我们不希望看到的，的确存在许多有限域，比如对一个素数 p，商环 $\mathbf{Z}/p$ 就是有 p 个元素的域。

在我们看到 $\mathbf{Z}/p$ 为域的证明之后，这应当就不那么让人惊讶了：我们已经知道 $\mathbf{Z}/p$ 是一个有单位元的交换环，并且是由理想 $p\mathbf{Z}$ 形成的 $\mathbf{Z}$ 的商环。所以只需要验证每个非零元素有一个乘法逆元即可。令 $x \in \mathbf{Z}/p$ 为非零元素，即对某个不被 p 整除的整数 y，$x = y + p\mathbf{Z}$。那么由欧几里得算法，存在整数 s,t，使

$$sy + tp = gcd(y, p) = 1$$

这样就有 $sy \equiv 1 \bmod p$。因此 $s + p\mathbf{Z}$ 就是 $x = y + p\mathbf{Z}$ 的乘法逆元。即任何非零元素都有一个乘法逆元，所以 $\mathbf{Z}/p$ 为域。

特别地，我们注意到对每个素数 p，的确存在包含 p 个元素的有限域。

另一方面，比如，不存在包含 6 个或 10 个元素的有限域。为什么？

尽管已经证明，存在包含素数方幂个元素的有限域。比如，9 个元素的有限域，128 个元素的有限域，但还需要做一些准备才能找出它们。

最简单的有限域就是有 p 个元素的域 $\mathbf{Z}/p$，p 为素数。由于许多不同的原因，我们需要比这更多的域。一个直接的原因是出于机器实现的考虑，使用特征为 2 的域是最方便的。在 $\mathbf{Z}/p$ 这样的域中，只有 $\mathbf{Z}/2$ 满足这个条件。同时，由于别的原因，我们希望域特别大。如果我们仅局限于 $\mathbf{Z}/p$ 上，就不可能同时满足这两方面的条件。

26.1 有限域的构造

有限域的构造以及有限域上的计算均建立在多项式计算的基础之上。为了描述简便，记 $\mathbf{F}_q$ 为有 q 个元素的有限域，当然对一个素数 p，$\mathbf{Z}/p = \mathbf{F}_p$。另外一个记号也比较常见，$\mathrm{GF}(q) = \mathbf{F}_q$，这里的 GF 代表**伽罗瓦域**。

对一个多项式 P（不必是不可约的）以及其他两个多项式 f 和 g，所有的系数均在 F_p 中，如果 $P \mid f - g$，则记 $f = g \bmod P$。这与整数的同余完全类似，由此还可定义：

$$\mathbf{F}_p[x]/P = \{\text{模}P\text{同余类}\}$$

这里一个多项式 f 的模 P **同余类** $\bar{f}$ 为 $\bar{f} = \{g \in \mathbf{F}_p[x] : g = f \bmod P\}$，通常将同余类仍记为 f。

一个多项式 f 是被**模** P **约简**的，如果 $\deg f < \deg P$。用多项式环 $\mathbf{F}_p[x]$ 上的除法算法，$\mathbf{F}_p[x]$ 中的每个多项式均模 P 等于一个模 P 约简的多项式。实际上，给定一个 f，由带余除法可以得到多项式 Q 和 R，且 $\deg R < \deg P$，使得

$$f = Q \cdot P + R$$

即 $f - R = Q \cdot P$，我们称 $f = R \bmod P$。

命题 两个模 P 约简的多项式 f 和 g 是模 P 相等的，当且仅当它们相等（在 $\mathbf{F}_p[x]$ 上）。

定理 对一个次数为 n 的不可约多项式 P，多项式模 P 的环 $\mathbf{F}_p[x]\bmod P=\mathbf{F}_p[x]/P$ 为一个**域**，且有 p^n 个元素。元素 $x-\text{mod}-P$ 是方程 $P(x)=0\bmod P$ 在 $\mathbf{F}_p[x]/P$ 中的一个根。

通常在 $\mathbf{F}_p[x]/P$ 上，希望用**约简的形式**表示任何元素，因为那样的话就容易验证两个元素是否相等：只需要比较系数。

设 k 是一个域，另一个域 K 包含 k，称 K 为 k 的一个**扩域**，而 k 则称为 K 的**子域**。k 的扩域 K 的**次数**就是用来构造 $K=k[x]/P$ 时所用多项式 P 的次数。

备注 这里，考虑 $\alpha=x-\text{mod}-P$ 为方程 $P(x)=0\bmod P$ 的一个根，我们称给 k **添加** $P(x)=0$ 的一个根，记为 $k[\alpha]=k[x]/P$。

习题

26.1.01 证明不存在元素 $x\in\mathbf{F}_{13}$，使 $x^5=1$，除非 $x=1$。

26.1.02 验证 x^2+x+1 是 $\mathbf{F}_2[x]$ 上惟一的不可约二次多项式。

26.1.03 验证 x^3+x+1 和 x^3+x^2+1 是 $\mathbf{F}_2[x]$ 上的两个不可约三次多项式。

26.1.04 验证 x^5+x+1 是 $\mathbf{F}_2[x]$ 上不可约。

26.1.05 验证在 $\mathbf{F}_2[x]$ 中，我们有特殊的等式 $(x^{l_1}+\cdots+x^{l_k})^2=x^{2l_1}+\cdots+x^{2l_k}$。

26.1.06 验证对素数 p，在 $\mathbf{F}_p[x]$ 上有特殊的等式

$$(x^{l_1}+\cdots+x^{l_k})^p=x^{pl_1}+\cdots+x^{pl_k}$$

26.2 域扩张的例子

本节我们将利用上一节的理论，给出几个具体的域扩张的例子。

例子 我们来看一下，在不用假设已经存在一个神秘的元素 $\sqrt{-1}$ 的情况下，如何构造复数域 $\mathbf{C}$，作为实数域 $\mathbf{R}$ 的一个**域扩张**。

首先我们证明 $x^2+1\in\mathbf{R}[x]$ 是不可约的。因为任何实数的平方都是非负的，因此方程 $x^2+1=0$ 在 $\mathbf{R}$ 中没有根。又因为 $x^2+1\in\mathbf{R}[x]$ 是二次的，如果它能在 $\mathbf{R}[x]$ 中分解，则必分解为两个线性因子的乘积（因为乘积的次数等于因子次数的和）。但是如果 x^2+1 有一个线性因子，则 $x^2+1=0$ 将在 $\mathbf{R}$ 中有一个根，而事实上它没有根。因此 x^2+1 在多项式环 $\mathbf{R}[x]$ 中是不可约的。

其次，由前面的讨论已知，$\mathbf{R}[x]\bmod I$ 是一个域，而且 $x^2=-1\bmod x^2+1$，故 $x-\text{mod}-(x^2+1)$ 就是 $\sqrt{-1}$。

我们也已证明，域扩张的任何元素 β 都可惟一地表示为 $\beta=a+b\alpha$ 的形式，这里的 a,b 均属于 $\mathbf{R}$。当然，我们通常记 x 在扩域中的像为 i，而不是 α。

例子 我们给域 $\mathbf{Z}/5$ 上添加一个 2 的平方根。首先注意到在 $\mathbf{Z}/5$ 中不存在 a 使 $a^2=5$，因此二次多项式 x^2-2 在 $\mathbf{Z}/5[x]$ 中不能分解（因为如果能够分解，则就会在 $\mathbf{Z}/5$ 中有一个根，而事实上没有），那么 $\mathbf{Z}/5[x]\bmod x^2-2$ 为一个域。在其中我们已把 $\mathbf{Z}/5$ 视为子域，并且 $x^2=2\bmod x^2-2$，所以 $x-\text{mod}-(x^2-2)$ 就是 2 的平方根。当然，通常将这个平方根记为 $\sqrt{2}$，而不是 α。

备注 这种构造可能被认为有些突然，因为构造过程中“构造”的多项式的根似乎不像

我们平常理解的根那样具体。但实际上，这种构造仍是相当直接的。

例子 我们给 $\mathbf{Z}/7$ 添加一个 2 的立方根。首先注意到 $\mathbf{Z}/7$ 中不存在 2 的立方根（可用穷举法验证，或者注意 $\mathbf{Z}/7^{\times}$ 是阶为 6 的循环群，由我们对循环 $\mathbf{Z}/7^{\times}$ 的了解，只有两个三次幂，它们是 ± 1），因此 2 不是立方根。

因此三次多项式 x^3-2 在 $\mathbf{Z}/7[x]$ 中不可约，如果它可约，则它必须有一个线性因子，也就是说 $x^3-2=0$ 在 $\mathbf{Z}/7$ 中必须有一个根，事实上它没有根。

由此，$(\mathbf{Z}/7)[x]-\mathrm{mod}-(x^3-2)$ 是一个域，且 $x-\mathrm{mod}-(x^3-2)$ 为 2 的一个立方根。$\mathbf{Z}/7$ 的域扩张中的每个元素 β 都可以惟一地表示为

$$\beta = a_0 + a_1\alpha + a_2\alpha^2$$

这里的 α 是 $x-\mathrm{mod}-(x^3-2)$ 的缩写。

习题

26.2.01 $\mathbf{Z}/7$ 中不存在 2 的三次方根，$\mathbf{Z}/7$ 中也不存在 3 的平方根。因此多项式 x^2-3 在 $\mathbf{F}_7[x]$ 中是不可约的，构造 $\mathbf{F}_{49}$ 作为 $\mathbf{F}_7[x]/(x^2-3)$ 的模型。在 $\mathbf{F}_{49}$ 中找出一个 2 的立方根。

26.2.02 $\mathbf{Z}/103$ 中不存在 2 的三次方根，$\mathbf{Z}/103$ 中也不存在 3 的平方根。因此多项式 x^2-3 在 $\mathbf{F}_{103}[x]$ 中是不可约的，构造 $\mathbf{F}_{103^2}$ 作为 $\mathbf{F}_{103}[x]/(x^2-3)$ 的模型。在 $\mathbf{F}_{103^2}$ 中找出一个 2 的立方根。

26.2.03 利用模型 $\mathbf{F}_{49}=\mathbf{F}_7[x]/(x^2-3)$，令 α 为 $x\bmod(x^2-3)$，α 是否为 $\mathbf{F}_{49}$ 的一个立方？$\alpha+1$ 是否也为一个立方？

26.2.04 利用模型 $\mathbf{F}_{49}=\mathbf{F}_7[x]/(x^2-3)$，令 α 为 $x\bmod(x^2-3)$，在 $\mathbf{F}_{49}$ 中找出一个 α 的五次根。

26.2.05 令 $P(x)=x^2-3$，不存在 3 模 5 的平方根，所以这个多项式在 $\mathbf{Z}/5$ 中没有根，因此它在 $\mathbf{F}_5[x]$ 中不可约。令 $K=\mathbf{F}_5[x]/P(x)$，并令 α 为 $x\bmod P(x)$，则 α 是 $\mathbf{F}_5$ 的扩域中 3 的一个平方根。在 K 中找出 x^2+x+1 的一个根。

26.2.06 设 K 同前，在 K 中找出 $x^4+1=0$ 的一个根。

26.2.07 令 $P(x)=x^2+1$，不存在 -1 模 7 的平方根，所以这个多项式在 $\mathbf{Z}/7$ 中没有根，因此它在 $\mathbf{Z}/7[x]$ 中不可约。令 $K=\mathbf{F}_7[x]/P(x)$，并令 α 为 $x\bmod P(x)$，则 α 是 $\mathbf{F}_7$ 的扩域中 -1 的一个平方根。求 $x^2-5=0$ 在 K 中的一个根。

26.2.08 设 K 同前，在 K 中找出 $x^{16}-x^8+1=0$ 的一个根。（这个多项式是 48 阶分圆多项式）

26.3 模 P 加法

有限域上的加法就是多项式的加法，它本身在结构上等同于向量的加法。

$\mathbf{F}_p[x]/P$ 上的加法很简单，只需要将多项式的对应系数相加即可。由于多项式的和的次数小于等于多项式的最大次数，因此两个约简多项式的和仍是约简的。

比如在 $\mathbf{F}_2[x]/(x^4+x+1)$ 中，将 x^3+x+1 与 x^2+x+1 相加将得到：

$$(x^3+x+1)+(x^2+x+1)=x^3+x^2+2x+2=x^3+x^2\bmod x^4+x+1$$

因为 2=0。

26.4 模 P 乘法

出于计算上的考虑，$\mathbf{F}_p[x]/P$ 上的乘法就是普通的多项式乘法，然后由模 P 约简。这与 $\mathbf{Z}/m$ 中的乘法非常相似。$\mathbf{F}_p[x]$ 上的乘法实际上有更本质更有意义的定义，它在一些定理证明中是必不可少的。但本节的这种形式只是出于计算的需要。

例子 在 $\mathbf{F}_2[x]/(x^4+x+1)$ 中，将 x^3+x+1 与 x^2+x+1 相乘将得到：

$$\begin{aligned}(x^3+x+1)\cdot(x^2+x+1) &= x^5+x^4+2x^3+2x^2+2x+1\\ &= x^5+x^4+1\\ &= x^2 \bmod x^4+x+1\end{aligned}$$

因为 2=0 且 $(x^5+x^4+1)-(x)(x^4+x+1)=x^2+1$。所以可用模 x^4+x+1 约简 x^5+x^4+1。

实际上，模 P 乘法就是将多项式以普通方式相乘后，再用模 P 约简。

习题

26.4.01 在域 $K=(\mathbf{F}_2)[x]/(x^2+x+1)$ 上，令 α 为 x 的像，以约简形式计算 $(1+\alpha)\alpha$。

26.4.02 在域 $K=(\mathbf{F}_2)[x]/(x^2+x+1)$ 上，令 α 为 x 的像，以约简形式计算 α^5。

26.4.03 在域 $K=(\mathbf{F}_2)[x]/(x^3+x+1)$ 上，令 α 为 x 的像，以约简形式计算 α^6。

26.4.04 在域 $K=(\mathbf{F}_2)[x]/(x^5+x^2+1)$ 上，令 α 为 x 的像，以约简形式计算 $(1+\alpha+\alpha^2)$ 与 $(1+\alpha^3)$ 的乘积。

26.5 模 P 乘法逆

这是最复杂的运算，需要使用欧几里得算法和快速指数算法。

这里很重要的一点就是模数 P 是不可约的。对于 $f\neq 0 \bmod P$，为了寻找 f 模 P 的逆元，就是使用扩展的欧几里得算法寻找多项式 S 和 T，使

$$S\cdot f+T\cdot P=1$$

那么即有

$$S\cdot f-1=T\cdot P$$

因此由定义有 $S\cdot f=1 \bmod P$，即 $f^{-1}=S \bmod P$。因为 $f\neq 0 \bmod P$ 且 P 是不可约的，必然它们的最大公约数为 1，所以这样的 S 和 T 是存在的。

比如，为在 $\mathbf{F}_2[x]/(x^2+x+1)$ 上求 x 的乘法逆元，首先做欧几里得算法：

$$(x^2+x+1)-(x+1)(x)=1$$

因此就得到了我们所期望的表达式：

$$(x+1)(x)+(1)(x^2+x+1)=1$$

由此即有

$$(x+1)(x)=1 \bmod (x^2+x+1)$$

即 $x^{-1}=x+1 \bmod (x^2+x+1)$。

为在 $\mathbf{F}_2[x]/(x^4+x+1)$ 上求 x^2+x+1 的乘法逆元，我们首先做欧几里得算法：

$$(x^4+x+1)-(x^2+x)(x^2+x+1)=1$$

因此得到我们所期望的表达式

$$(x^2+x)(x^2+x+1)+(1)(x^4+x+1)=1$$

由此，

$$(x^2+x)(x^2+x+1)=1 \bmod (x^4+x+1)$$

这样就有

$$(x^2+x+1)^{-1}=x^2+x \bmod (x^4+x+1)$$

习题

26.5.01 在域 $K=\mathbf{F}_2[x]/(x^2+x+1)$ 上，令 α 为 x 的像，以约简形式计算 $(1+\alpha)^{-1}$。

26.5.02 在域 $K=\mathbf{F}_2[x]/(x^2+x+1)$ 上，令 α 为 x 的像，以约简形式计算 α^{-1}。

26.5.03 在域 $K=\mathbf{F}_2[x]/(x^3+x+1)$ 上，令 α 为 x 的像，以约简形式计算 $(1+\alpha+\alpha^2)^{-1}$。

26.5.04 验证多项式 x^4+x+1 在 $\mathbf{F}_2[x]$ 上不可约，令 α 为 $x^4+x+1=0$ 的根，用 α 的多项式形式表示 $(\alpha^2+1)^{-1}$。

26.5.05 多项式 x^6+x^3+1 在 $\mathbf{F}_2[x]$ 上不可约，令 α 为 $x^6+x^3+1=0$ 的根，用 α 的多项式形式表示 $(\alpha^3+\alpha+1)^{-1}$。

26.5.06 多项式 $x^6+x^5+x^4+x^2+1$ 在 $\mathbf{F}_2[x]$ 上不可约，α 为 $x^6+x^5+x^4+x^2+1=0$ 的根，用 α 的多项式形式表示 $(\alpha^3+\alpha+1)^{-1}$。

第27章 离散对数

回想一下，a的以b为底模p的**离散对数**L（如果存在），就是满足$a=b^L$的L，我们记为$L=\log_b a$，模数的理解参照上下文。为了明确这个对数不是“通常的”对数，L有时也称为a的以b为底模p的指标。我们已经知道对素数p，存在一个模p的本原根b。因此有一个模素数p的本原根b，任何模p的非零元素a都有一个以b为底的离散对数。对这一思想的抽象，通过利用循环群的生成元代替$\mathbf{Z}/p^{\times}$中的本原根b的方式实现。这种抽象之所以被发现很有用处，是因为计算离散对数的最快速算法只适用于简单的情况，如$\mathbf{Z}/p^{\times}$，但不适用于像椭圆曲线这样的机制。

在一个群上计算离散对数最直接的方法就是穷举，简单地试验每个元素，直到找出正确的那一个。即在$\mathbf{Z}/p$中，b为模p的本原根，为计算$\log_b a$，首先计算$b^1,b^2,b^3,b^4,\cdots$（均模p），直到出现b的某个方幂为目标a。这需要$O(p)$次试验，对于较大的p，比如用于密码目的的p（100位十进制数或更大），这种天真的方法是完全不可行的，就像试除法在整数分解中不可行一样。

27.1 Baby-step Giant-step 算法

计算离散对数的Baby-step Giant-step算法在任何循环群G上都可运用，条件是群运算的计算代价不是太昂贵。运行时间是$|G|$平方根阶的，即为$O(\sqrt{|G|})$。另一方面，该算法要使用$O(\sqrt{|G|})$个存储器。因此它是运行时间和存储器之间平衡较好的一个算法。该算法是确定性的。

设G是以b为生成元的循环群，我们希望计算某个其他元素$a\in G$以b为底的离散对数。令n为群G的阶$|G|$，并令$m=\text{floor}(\sqrt{n})$，即m为恰好不超过$\sqrt{n}$的整数。当$l=\log_b a$时，我们当然可以记$l=m\cdot i+j$，其中$0\leqslant i<m$，$0\leqslant j<m$。方程$a=b^{mi+j}$可变形为$a(b^{-m})^i=b^j$。这样我们首先对$0\leqslant j<m$，计算m个b^j，并把它们放在某个有序的查询表中。然后连续计算如下m个值：

$$a,a\cdot b^{-m},a\cdot b^{-2m},\cdots,a\cdot b^{-mi},\cdots$$

在第i步，比较$a\cdot b^{-mi}$和b^j，如果相等，则$\log_b a=mi+j$。

备注 关于可查询表b^j的构造，最初需要$\sqrt{n}$步。$a\cdot b^{-im}$和b^j的比较共有$O(\sqrt{n})$次，每一次应当必须在不超过$O(\sqrt{n})$的时间内完成。否则总共就需要$O(n)$步，这就与穷举法没有区别了。因此查询表b^j的编排必须保证能非常快速地判断一个元素$a\cdot b^{-im}$是否在表中。比如，在简单的情况，这个判断应只需要$\log_2 m$次比较即可完成。

尽管这个算法在任何群上均有意义，但b^j的必要排序最容易在有形的具体例子上描述，如$\mathbf{Z}/p^{\times}$，p为素数。设b是模素数p的一个本原根，群$G=\mathbf{Z}/p^{\times}$的阶为$n=p-1$，设m是不超过$\sqrt{p-1}$的最大整数。为在群G中计算$\log_b a$,首先计算

$$(0,b^0\%p),(1,b^1\%p),(2,b^2\%p),\cdots,(m-1,b^{m-1}\%p)$$

我们将这些有序对按它们第二个分量的大小进行排序，所以检验某个 $g \in G$ 是否在表上，就仅需要 $\log_2 m$ 次比较。那么可以对比较计算步骤描述如下：

- 计算 $c = b^{-m} \bmod p$；
- 初始化 $x = a$；
- 对 $0 \leqslant i < m$，如果 x 是表 $b^0, b^1, \cdots, b^{m-1}$ 中的 b^j，则 $\log_b a = mi + j$。否则，用 $x \cdot c \bmod p$ 代替 x，且令 $i=i+1$。

备注 为使计算更加有效有更方便，G 中元素的大小或其他数字标识的记法是必要的。

例子 计算 $\log_2 3 \bmod 29$。这里 $p=29$，$a=3$，$b=2$，$m = \text{floor}(\sqrt{29-1}) = 5$。因此我们需要计算 $(0,b^0\%29),(1,b^1\%29),(2,b^2\%29),(3,b^3\%29),(4,b^4\%29)$，即得到这样一个表：(0,1),(1,2),(2,4),(3,8),(4,16)。因为这个例子中的数小，得到的这个表已经是按照数对的第二个分量由小到大排列了。令 $c = 2^{-5} \bmod 29 = 10$，初始化 $x = a = 3$，它不在表中；用新的值 $3 \cdot 2^{-5} = 2^0$ 取代 x=3，重新在表中找 2^0，这一次找到了 2^0，此时 $i=1$，$j=0$，所以 $\log_2 3 = 5 \cdot 1 + 0 = 5$。

例子 计算 $\log_2 3 \bmod 101$。这里 $p=101$，$a=3$，$b=2$，$m = \text{floor}(\sqrt{101-1}) = 10$。因此我们需要计算 $(0,b^0\%101),(1,b^1\%101),\cdots,(9,b^9\%101)$，即得到这样一个表：(0,1),(1,2),(2,4),(3,8),(4,16),(5,32),(6,64),(7,27),(8,54),(9,6)。按照数对的第二个分量由小到大排列则得到

$$(0,1),(1,2),(2,4),(9,6),(3,8),(4,16),(7,27),(5,32),(8,54),(6,64)$$

同样，令 $c = 2^{-9} \bmod 101 = 29$，初始化 $x = a = 3$，发现 3 不在表中；用 $x \cdot c \bmod p$ 代替 x，并且只要 x 不在表中就继续计算：

$$3 \cdot 29\%101 = 87$$
$$87 \cdot 29\%101 = 99$$
$$99 \cdot 29\%101 = 43$$
$$43 \cdot 29\%101 = 35$$
$$35 \cdot 29\%101 = 5$$
$$5 \cdot 29\%101 = 44$$
$$44 \cdot 29\%101 = 64$$

这一次我们发现 64 在表中，即 $3 \cdot (2^{-9})^7 = 64 = 2^6 \bmod 101$，此时 $i=7$，$j=6$，所以 $\log_2 3 = 7 \cdot 9 + 6 = 69$ 在 $\mathbf{Z}/101^{\times}$ 中。

习题

27.1.01 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/29$ 中求 $\log_2 3$。
27.1.02 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/29$ 中求 $\log_2 5$。
27.1.03 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/29$ 中求 $\log_2 7$。
27.1.04 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/29$ 中求 $\log_2 11$。
27.1.05 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/53$ 中求 $\log_2 3$。
27.1.06 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/53$ 中求 $\log_2 5$。
27.1.07 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/53$ 中求 $\log_2 7$。
27.1.08 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/53$ 中求 $\log_2 11$。

27.1.09 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/227$ 中求 $\log_2 3$。

27.1.10 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/227$ 中求 $\log_2 5$。

27.1.11 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/227$ 中求 $\log_2 7$。

27.1.12 利用 Baby-step Giant-step 算法，在 $\mathbf{Z}/227$ 中求 $\log_2 11$。

27.2 Pollard 的 Rho 方法

分解整数的 Pollard 的 Rho 算法的思想可用于计算任意群上的离散对数。在因式分解中，Rho 方法比直接的天真方法获得了极好的速度性能，但还不是多项式时间的。这里的 Rho 算法本质上是概率意义上的，依赖于生日悖论。在这方面，它与分解整数的 Rho 算法完全平行。

备注 本算法可用来计算任意有限循环群上的离散对数，不仅仅是对 $\mathbf{Z}/p$ 或其他情况。比如它可用于计算椭圆曲线上的离散对数。

备注 当所用的循环群的阶为素数时，这个算法最容易实现。因此，比如考虑在 $\mathbf{Z}/59^{\times}$ 上的离散对数问题时，由于 $\mathbf{Z}/59^{\times}$ 的阶为 58 且有两个生成元，我们不用直接去考查 $\mathbf{Z}/59^{\times}$，而只需利用该算法的简单形式，去考查 29 阶循环子群 G，G 由 $b=4=2^2$ 生成。那么我们只需对那些小的子群 G 中的元素计算离散对数（子群 G 包含 $\mathbf{Z}/59^{\times}$ 的平方）。

设 G 为一个循环群，生成元为 b，单位元为 e。对于 $a\in G$，我们来计算 $\log_b a$，也就是要寻找最小的正整数 L，使 $b^L=a$。将 G 划分为三个不相交的子集 S_1、S_2 和 S_3，且 $e\notin S_2$。定义 G 到 G 的“随机”函数 f 如下：

$$f(X)=\begin{cases} a\cdot X & \text{如果} X\in S_1 \\ X^2 & \text{如果} X\in S_2 \\ b\cdot X & \text{如果} X\in S_3 \end{cases}$$

选择两个随机整数 m_0 和 n_0（可能都为 0），令 $X_0=a^{m_0}\cdot b^{n_0}$，递归地定义一个元素 $X_i\in G$ 的序列，$X_{i+1}=f(X_i)$。

在分解整数的 Rho 方法中运用的概率机制同样在这里起作用。假设序列 $X_0,X_1,\cdots$ 是“随机的”，G 的阶为 $|G|$，则对于 $N>\sqrt{|G|}$，$X_i=X_j$ 的概率至少为 $\frac{1}{2}$，这里 $i\neq j$ 且 $i,j\leqslant N$。

同 Rho 整数分解方法一样，我们也使用 Floyd 循环检测方法，以避免使用太多的存储器和进行大量的比较操作。为达到这个目的（同时计算 X_i 的序列），我们计算另一个序列 $Y_0,Y_1,Y_2,\cdots$，它的定义如下：

$$Y_0=e,\quad Y_{i+1}=f(f(X_{2i}))$$

因此，$Y_i=X_{2i}$。那么对刚好大于 $\sqrt{|G|}$ 的 N，$Y_i=X_i$，即 $X_{2i}=X_i$ 的概率大于 $\frac{1}{2}$。

由我们的构造，每个 X_{i+1} 都可以表示为 a 的方幂和 b 的方幂的乘积，因为函数 f 完全是按照 a 或 b 的乘法以及平方来定义的。为了形成这种概念，我们令 m_i 和 n_i 是正整数且使得 $X_i=a^{m_i}b^{n_i}$，m_0 和 n_0 是一开始就选定了的（前面已经指出，m_0 和 n_0 都为 0 是一种合理的选择）。如果我们要清楚地说明 m_i 和 n_i 是怎样由函数 f 的定义递归地确定的，那么就是：

$$m_{i+1}=\begin{cases} m_i+1 & \text{如果} X_i \in S_1 \\ 2m_i & \text{如果} X_i \in S_2 \\ m_i & \text{如果} X_i \in S_3 \end{cases}$$

$$n_{i+1}=\begin{cases} n_i & \text{如果} X_i \in S_1 \\ 2n_i & \text{如果} X_i \in S_2 \\ n_i+1 & \text{如果} X_i \in S_3 \end{cases}$$

那么当 $X_{2i}=X_i$ 时，我们就可以得出 $a^{m_{2i}}b^{n_{2i}}=a^{m_i}b^{n_i}$，由此即得 $b^{n_{2i}-n_i}=a^{m_i-m_{2i}}$，两同时取以 b 为底的对数，则得到：

$$(\log_b a)(m_{2i}-m_i)=(n_i-n_{2i}) \bmod |G|$$

这是因为由生成元的定义，$|b|=|G|$。除非我们碰到 $m_{2i}-m_i$ 与阶 $|G|$ 不是互素的情况，否则总能找到 $m_{2i}-m_i$ 模 $|G|$ 的乘法逆元 t，那么所要求的离散对数则为

$$\log_b a=t(n_i-n_{2i}) \quad \bmod |G|$$

因此这个算法需要跟踪六个量，即 $(X_i,m_i,n_i,Y_i,m_{2i},n_{2i})$，$i=0,1,2,\cdots$，到 $X_i=Y_i$ 时终止。为了给出一个更适合于执行的过程，定义函数 $F:G\times\mathbf{Z}\times Z\to G\times\mathbf{Z}\times Z$ 为：

$$F(X,m,n)=\begin{cases} (a\cdot X,m+1,n) & \text{如果} X_i \in S_1 \\ (X^2,2m,2n) & \text{如果} X_i \in S_2 \\ (b\cdot X,m,n+1) & \text{如果} X_i \in S_3 \end{cases}$$

为了使算法描述和符号更加简洁，算法按如下步骤执行：

- 选取随机整数 m_0 和 n_0；
- 初始化 $(X,m_X,n_X,Y,m_Y,n_Y)=(a^{m_0}b^{n_0},m_0,n_0,a^{m_0}b^{n_0},m_0,n_0)$；
- 重复如下步骤直到 $X=Y$：用 $F(X,m_X,n_X)$ 代替 (X,m_X,n_X)，并且用 $F(F(Y,m_Y,n_Y))$ 代替 (Y,m_Y,n_Y)；
- 当 $X=Y$ 时，尝试求解如下方程：$(\log_b a)(m_Y-m_X)=(n_X-n_Y) \quad \bmod |G|$。

如果 m_Y-m_X 与 $|G|$ 不是互素的，则算法失败。假设 $|G|$ 为素数，这样就方便多了：如果我们想象整数 $m_{2i}-m_i$ 是“随机的”，则它与 $|G|$ 有一个公因子的概率大约就是 $1/|G|$。对于较大的 $|G|$，这个概率值是可以忽略的。一旦算法失败，重新选择 m_0 和 n_0，再次执行算法。

备注 如果群 G 的阶 $|G|$ 不是素数，特别地，它有一个小的素因子 p，由于 m_Y-m_X 与 $|G|$ 的最大公因子就等于 p，则算法失败概率至少为 $1/p$，这就不能忽视了。但这可以通过多次选择随机种子 m_0 和 n_0 并重复执行算法的方法来处理。

备注 选择子集 S_i 时，必须保证在判断一个给定的元素 $X\in G$ 是否属于子集 S_i 时的效率足够高。

备注 设 G 是 $\mathbf{Z}/59^{\times}$ 的一个子群，且包含模 59 的平方。由循环群的一般结论，因为 2 是模 59 的本原根，4 就是 G 的一个生成元。即任何模 59 的非零的平方，就有一个以 4 为底的离散对数。另外，子群 G 的阶为 29 是个素数，我们知道 9 是模 59 的一个平方，所以 9 属于 G。我们来计算 $\log_4 9$。首先必须确定如何选择子集 S_1、S_2 和 S_3，一种通用的方法是选取：

$$S_1=G\cap\{1,4,7,10,\cdots,55,58\}$$

$$S_2 = G \cap \{2,5,8,11,\cdots,56\}$$
$$S_3 = G \cap \{3,6,9,12,\cdots,57\}$$

选取 $m_0 = n_0 = 0$，所以 $X_0 = Y_0 = 1 \bmod 59$，然后做初始化如下：

$$(X, m_X, n_X) = (1,0,0)$$
$$(Y, m_Y, n_Y) = (1,0,0)$$

且 $a = 9$，$b = 4$。下面开始迭代步骤，每一步的六元组依次为：

X	m_X	n_X	Y	m_Y	n_Y
1	0	0	1	0	0
9	1	0	36	1	1
36	1	1	27	2	4
26	1	2	28	3	5
27	2	4	26	5	5
49	2	5	49	10	11

在第五步我们发现 $X = Y$，所以我们就有 $(\log_4 9)(10-2) = (5-11) \bmod 29$（注意，我们解这个方程是模 29=（59 - 1）/ 2，而不是 59），这样在 $\mathbf{Z}/59^{\times}$ 中，$\log_4 9 = 21$。

例子 S_i 的选取方法同上，我们来计算模 227 的离散对数，之所以选取这个模数，是因为它和 $113 = (227-1)/2$ 都是素数。正好 2 为模 227 的一个本原根，所以 $4 = 2^2$ 就是 $\mathbf{Z}/227^{\times}$ 的一个阶为 113 的循环子群 G 的生成元。因此，我们可以计算任何模 227 非零平方的以 4 为底的离散对数。比如，$17^2 \% 227 = 62$，所以 62 就会有模 227 以 4 为底的离散对数。对种子 m_0 和 n_0 做最简单的选择，即 $m_0 = n_0 = 0$，所以 $X_0 = Y_0 = 1 \bmod 227$，其他初始值为 $a = 62$，$b = 4$，$(X, m_X, n_X) = (1,0,0)$，$(Y, m_Y, n_Y) = (1,0,0)$，迭代运算开始，各组数对的计算结果如下：

X	m_X	n_X	Y	m_Y	n_Y
1	0	0	1	0	0
62	1	0	212	2	0
212	2	0	219	4	1
225	4	0	99	4	3
219	4	1	36	5	4
195	4	2	122	5	6
99	4	3	62	10	13
169	4	4	225	40	52
36	5	4	195	40	54
144	5	5	169	40	56
122	5	6	144	41	57
129	10	12	129	82	116

我们发现在第 11 步出现 $X = Y$，因此通过求解如下方程

$$(\log_4 62) \cdot (82-10) = 12 - 116 \bmod 113$$

即得 $\mathbf{Z}/227^{\times}$ 上以 4 为底 62 的离散对数 $\log_4 62 = 99$。

为了给出另外的例子，我们需要一个有限域的模型。多项式 $P(X) = X^5 + X^2 + 1$ 在 $\mathbf{F}_2[X]$ 上是不可约的，这可由试除法验证如下：我们只需用次数不超过 P 的次数的一半的不可约多项式来做除法，那么这里就需要用 X、$X+1$ 和 X^2+X+1 来做除法，它们没有一个能整除 P。因此 $K = \mathbf{F}_2[X]/P$ 是一个有 $2^5 = 32$ 个元素的有限域。因为 P 的次数为 5，而且 $2^5 - 1 = 31$ 为素数，X 就是 K 中模 P 的一个本原根。因此，我们可以计算 K 中以 X 为底模 P 的对数。注意，我们使用大写字母“X”表示多项式中的不定元，以区别 Rho 算法中的记号“x”。

备注 注意 $2^5 - 1 = 31$ 为素数，所以可以运用 Rho 算法的最直接的形式。

例子 $P(X) = X^5 + X^2 + 1$，我们来计算 $K = \mathrm{F}_2[X]/P$ 中以 X 为底模 P 的离散对数。我们需要从 K 的 31 个非零元素中选取三个集合 S_1、S_2 和 S_3，而且要保证验证一个模 P 的多项式属于某个集合 S_i 是容易的。尝试模拟“随机性”的选择：

- 取 S_1 为包含次数不超过 5 且 x^2 项的系数等于 x^4 项系数的多项式；
- 取 S_2 为包含次数不超过 5 且 x^3 项的系数等于常数项和线性项系数之和的多项式；
- 取 S_3 为包含次数不超过 5 且不在集合 S_1 和 S_2 中的多项式。

由上面的记号，$b = x, a = x+1$，对任何的多项式 X，定义

$$F(X,m,n) = \begin{cases} ((x+1)\cdot X, m+1, n) & \text{如果} X \in S_1 \\ (X^2, 2\cdot m, 2\cdot n) & \text{如果} X \in S_2 \\ (x\cdot X, m, n+1) & \text{如果} X \in S_3 \end{cases}$$

同样取最简单的随机种子，$m_0 = n_0 = 0$，初始化

$$(X, m_X, n_X, Y, m_Y, n_Y) = (1,0,0,1,0,0)$$

求 $F(X, m_X, n_X)$ 和 $F(F(Y, m_Y, n_Y))$，直到 $X = Y$：

1	0	0	1	0	0
$1+x$	1	0	$1+x^2$	2	0
$1+x^2$	2	0	x^2+x^3	8	0
$1+x^4$	4	0	$1+x^4$	16	1

现在只需要解如下方程即可：$(\log_x(x+1))\cdot(16-4) = 0 - 1 \bmod 31$，由此，我们可得（模 x^5+x^2+1）$\log_x(x+1) = 18$。

备注 如果子集 S_i 的选取“不是太随机”或者等价地，f“不是太随机”，则需要很多步才能得到 $X = Y$。

例子 条件同前例，S_i 的选取为：

- 取 S_1 为包含次数不超过 5 且 x^2 项的系数等于 x^4 项系数的多项式；
- 取 S_2 为包含次数不超过 5 且 x^3 项的系数等于常数项和线性项系数之和的多项式；
- 取 S_3 为包含次数不超过 5 且不在集合 S_1 和 S_2 中的多项式。

则算法运行如下：

1	0	0	1	0	0
x	0	1	$x+x^2$	1	1
$x+x^2$	1	1	$x+x^2+x^3+x^4$	3	1
$x+x^3$	2	1	$1+x^2+x^4$	8	2
$x+x^2+x^3+x^4$	3	1	$1+x+x^2+x^3+x^4$	16	6
$1+x+x^2$	4	1	$1+x+x^4$	33	12
$1+x^2+x^4$	8	2	x^3+x^4	67	24
$1+x+x^2+x^3$	8	3	$x+x^3+x^4$	68	25
$1+x+x^2+x^3+x^4$	16	6	$1+x+x^2+x^3$	138	50
$x+x^4$	32	12	$x+x^4$	552	200

尽管我们还是得到了相同的结果（离散对数 $\log_x(x+1)=18$），但它用了 9 步而不是前面的 3 步。

例子 用相同有限域模型 $GF(2^5)$，我们来计算 $\log_x(x^2+x+1) \bmod P$。S_i 的选取与第一个有限域的例子中的相同。详细过程如下：

1	0	0	1	0	0
$1+x+x^2$	1	0	$1+x^2+x^4$	2	0
$1+x^2+x^4$	2	0	x^3	3	1
x^2	3	0	$1+x^3+x^4$	5	1
x^3	3	1	$x+x^3$	6	2
$1+x^2+x^3+x^4$	4	1	x^3	14	4
$1+x^3+x^4$	5	1	$1+x^3+x^4$	16	4

因此，通过解如下方程

$$(\log_x(x^2+x+1))\cdot(16-5)=1-4 \bmod 31$$

我们就得到 $\log_x(x^2+x+1)=11$。

习题

27.2.01 用 Rho 方法求 $\mathbf{Z}/23^{\times}$ 中以 2 为底 3 的对数。

27.2.02 用 Rho 方法求 $\mathbf{Z}/23^{\times}$ 中以 2 为底 6 的对数。

27.2.03 用 Rho 方法求 $\mathbf{Z}/23^{\times}$ 中以 2 为底 13 的对数。

27.2.04 用 Rho 方法求 $\mathbf{Z}/23^{\times}$ 中以 2 为底 12 的对数。

27.2.05 用 Rho 方法求 $\mathbf{Z}/47^{\times}$ 中以 2 为底 9 的对数。

27.2.06 用 Rho 方法求 $\mathbf{Z}/47^{\times}$ 中以 2 为底 3 的对数。

27.2.07 用 Rho 方法求 $\mathbf{Z}/47^{\times}$ 中以 2 为底 6 的对数。

27.2.08 用 Rho 方法求 $\mathbf{Z}/47^{\times}$ 中以 2 为底 21 的对数。

27.2.09 用 Rho 方法求 $\mathbf{Z}/2027^{\times}$ 中以 3 为底 10 的对数。

27.2.10 用 Rho 方法求 $\mathbf{Z}/2027^{\times}$ 中以 3 为底 4 的对数。

27.2.11 用 Rho 方法求 $\mathbf{Z}/2027^{\times}$ 中以 3 为底 12 的对数。

27.2.12 用 Rho 方法求模 x^5+x^2+1 以 x 为底 x^3+x^2+x+1 的对数。

27.2.13 用 Rho 方法求模 x^5+x^2+1 以 x 为底 x^4+x+1 的对数。

27.2.14 用 Rho 方法求模 x^5+x+1 以 x 为底 $x^4+x^3+x^2+x+1$ 的对数。

27.2.15 用 Rho 方法求模 x^5+x^2+1 以 x 为底 x^4+x^3+x+1 的对数。

27.2.16 用 Rho 方法求模 x^7+x^3+x+1 以 x 为底 $x+1$ 的对数。

27.2.17 用 Rho 方法求模 x^7+x^3+x+1 以 x 为底 x^2+1 的对数。

27.2.18 用 Rho 方法求模 x^7+x^3+x+1 以 x 为底 $x^5+x^4+x^3+x^2+x+1$ 的对数。

27.2.19 用 Rho 方法求模 x^7+x^3+x+1 以 x 为底 x^5+x^4+1 的对数。

27.3 指数演算

本节的方法，**指数演算**方法，是计算循环群 G 上离散对数的已知最好的方法。这个算法不适用于所有循环群，但如果它能适用某个群，则它是一个子指数时间算法。该算法是概率意义上的算法。

备注 在一开始做一个声明是必要的，那就是：算法中所用到的 G 的子集 S，即所谓的**因子基**，必须很好地选择。但是一般情况下，又不明确如何做一个好的选择，甚至是不是存在一个好的选择。调整算法可以达到对因子基选择的校正，但通常可能不存在一个足够好的选择，以获得子指数的运行时间。

设 b 是循环群 G 的生成元，$n=|G|$，并设 $a\in G$ 是另一个元素，我们来计算 a 的以 b 为底的对数。

- 选择因子基：选择 G 的一个较小的子集，**因子基** $S=\{p_1,\cdots,p_t\}$；
- 收集关系：选择一个随机整数 k，$0\leqslant k\leqslant n-1$，尝试用 S 中一组元素的乘积来表示 b^k。如果这种关系 $b^k=p_1^{k_1}\cdots p_t^{k_t}$ 存在，两边取对数即得
$$k=k_1\log_b p_1+\cdots+k_t\log_b p_t \bmod n$$
重复这一过程，直到有超过 t 个关系。
- 求 $\log_b p_i$：求解方程组以求得因子基中元素以 b 为底的对数。如果方程组是不定的，则返回到上一步并生成更多的关系。
- 求另一个对数：给定 $a\in G$，选择一个随机整数 k，并尝试将 $a\cdot b^k$ 表为如下形式：
$$a\cdot b^k=p_1^{k_1}\cdots p_t^{k_t}$$
如果成功，求解 $\log_b a=(k_1\log_b p_1+\cdots+k_t\log_b p_t)-k \bmod n$；
如果不成功，选择不同的 k，并返回重新计算。

备注 当然，计算群中不同元素的对数时，对因子基元素的对数的预先计算，不必要重复。

备注 尽管算法的检验已解释了，但因子基 S 的选择必须保证 G 中的“许多”元素能被表示为 S 中的元素的指数“较小”方幂的乘积。当然，这个思想及其影响需要量化以优化算法。

备注 当所考虑的循环群是 $G=\mathbf{Z}/p^{\times}$，p 为素数，则因子基就是前 t 个素数。当然，选择一个合适的 t 却是一个问题。检验一个随机产生的群的元素是否为因子基中元素的乘积（以较小的正整数指数），可以用因子基中元素的试除法很快完成。

例子 设 $G=\mathbf{Z}/29^{\times}$，G 的阶为 28，选择本原根 $b=11$。取因子基 $S=\{2,3,5\}$，考虑 b 的随机方幂，开始生成关系。注意我们要解由模 28 的关系组成的方程组：

$$b^2\%29=11^2\%29=5$$

$$b^3\%29 = 11^3\%29 = 26 \text{（失败，不能表为因子基元素的乘积）}$$
$$b^5\%29 = 11^5\%29 = 14 \text{（失败，不能表为因子基元素的乘积）}$$
$$b^6\%29 = 11^6\%29 = 9 = 3^2$$
$$b^7\%29 = 11^7\%29 = 12 = 2^2 \cdot 3$$
$$b^9\%29 = 11^9\%29 = 2$$

恰好在本例中，可以通过解模 28 的方程组获得因子基的对数。由第一个关系式就可直接得到 $\log_{11} 5 = 2$。由第四个关系式则可得 $6 = 2 \cdot \log_{11} 3 \bmod 28$，因为对数前的系数 2 与模数 28 有一个公因子，因此不能惟一地确定 $\log_{11} 3$。但是由最后一个关系式可直接得到 $\log_{11} 2 = 9$。然后我们再利用倒数第二个关系式，则有 $7 = 2 \cdot \log_{11} 2 + \log_{11} 3 \bmod 28$，可得 $\log_{11} 3 = 17$。

这就完成了对因子基 $S = \{2,3,5\}$ 的预先计算。比如，为了求 $\log_{11} 7$，用 $b = 11$ 的“随机”方幂乘以 $a = 7 (\bmod 29)$，然后寻找可表示为因子基元素乘积的结果：

$$a \cdot b\%29 = 7 \cdot 11\%29 = 19 \text{ (失败)}$$
$$a \cdot b^2\%29 = 7 \cdot 11^2\%29 = 6 = 2 \cdot 3$$

因此，

$$\log_{11} 7 = \log_{11} 2 + \log_{11} 3 - 2 = 9 + 17 - 2 = 24$$

备注 该算法比前面仅适用较大群的算法效率更高。

第28章 椭圆曲线

有限域上的椭圆曲线，给出了这样一种群的例子，即该群的结构甚至比 $\mathbf{Z}/p^{\times}$ 的结构还要简单。后面我们将给出椭圆曲线的简短定义。这也为**抽象** Elgamal 密码的思想来编制密码提供了可能，Elgamal 密码使用了群 $\mathbf{Z}/p^{\times}$ 上的离散对数。

在离散对数类密码中，按照可对抗密钥长度攻击的能力而言，椭圆曲线密码似乎是目前最优的，例如与使用群 $\mathbf{Z}/p^{\times}$ 的 Elgamal 密码相比。当然这也可能随着人们开发出适合于椭圆曲线的更好算法而改变。

而且似乎有事实表明，椭圆曲线（离散对数）密码较 RSA 体制能更好地对抗密钥长度的攻击。这也可能随着人们开发出适合于椭圆曲线的更好算法而改变。

也有一种基于如下思想的**因子分解攻击**，即如果 $\mathbf{Z}/n$ 不是一个域，也就是 n 不是一个素数，则群运算公式可能会在不同的点“爆炸”。对于特定范围的大整数，这种分解攻击是凑效的。尽管目前在其优化范围内还不如二次筛法，也不如针对真正大整数的数域筛法。但是在实践中椭圆曲线筛法仍被广泛使用。

本章节仅仅给出椭圆曲线结构的一般介绍。最近有一本书解释了有关密码学的椭圆曲线上的计算问题（当然还包括其他计算问题），它就是参考文献中所列的[Blake,Seroussi,Smart 2000]。从减少计算量的角度出发，更多地强调了代数几何的系统化运用，[Silverman 1986]利用丰富的高等代数和代数几何的理论，给出了椭圆曲线一般理论的全面且系统化的介绍。

28.1 抽象的离散对数

最简单的情况就是我们已知的 $G=\langle b\rangle$，毫无疑问，G 中的每个元素都有一个离散对数。

还有一种情况就是 $G=\mathbf{Z}/n$，这里的运算为模 n 加法，且是 b 与 n 互素的。因为群的运算为加法，指数运算（反复的群运算）就正好是乘法。因此这里的离散对数就是

$$\log_b x = l \text{ 且使 } l\cdot b = x$$

这种简单的情况容易计算 x 的离散对数为 $\log_b x = b^{-1}\cdot x$。我们感兴趣的则是那些计算离散对数比较**难**的情况。

首先使用离散对数的情况就是在群 $\mathbf{Z}/p^{\times}$ 上（模 p 乘法），p 为素数。我们选择模 p 的**本原根** b 为底，它是存在的：即 b 就是使得 $\mathbf{Z}/p^{\times}$ 的每个非零元素均能表示为 b 的方幂的元素。实际上，$\mathbf{Z}/p^{\times}$ 中存在 $\varphi(p-1)$ 个本原根，这里的 φ 是欧拉函数。（在有些情况底数 b 并不一定是本原根，因为有时需要对 b 的方幂取对数。）

不太严格地讲，给定一个“大的”素数 p 以及 b 和 x，似乎计算 $\log_b x$ 的难度相当于分解与 p 大小相当的整数 N 的难度。

28.2 离散对数

下面我们来看一下 **Elgamal 密码**的抽象形式。首先回忆一下该密码是如何工作的：给定

一个大素数 $p>10^{150}$，一个模 p 的本原根 b（即意味着任何 y 都可表示为 $y=b^L \bmod p$），一个整数 c，$1<c<p$。秘密就是方幂 l（**离散对数**）使得 $b^l=c \bmod p$，只有解密者知道 l，解密者选择 l 并计算 c^l。

加密步骤如下：将明文 x 编码为一个整数，使 $0<x<p$，加密者选取一个辅助的随机数 r，这个随机数只是为加密者所知的一个临时秘密。x 的加密方式如下：

$$y=E_{b,c,p,r}(x)=(x\times c^r)\%p$$

随同这个加密消息发送的还有“头部” b^r。注意加密者只需知道 b,c,p，并选择随机数 r，但不知道离散对数 l。

解密步骤则需要离散对数 l 的信息，但不需要随机数 r。首先由“头部” b^r，解密者计算

$$(b^r)^l=b^{r\cdot l}=(b^l)^r=c^r \bmod p$$

然后给上面等式两边同时乘以 c^r 模 p 的乘法逆元 $(c^r)^{-1}$，即可恢复明文消息：

$$D_{b,c,p,r,l}(y)=(c^r)^{-1}\cdot y=(c^r)^{-1}\cdot c^r\cdot x=x \bmod p$$

抽象形式。现在我们对 Elgamal 算法进行抽象以使它可用于任意的群，设 G 为一个群，给定 $b\in G$。这里的群 G **不必是**由 b 生成的循环群。

选取一个秘密整数 l，并设 $c=b^l$，只有解密者知道 l，它是 c 在 G 中的以 b 为底的**离散对数**。

备注 我们不是先选择 c 然后去计算它的离散对数！

加密步骤如下：将明文元素 x 编码为元素 $x\in G$。加密者选取一个辅助的随机数 r，这个随机数只是为加密者所知的一个临时秘密。x 的加密方式如下：

$$y=E_{b,c,p,r}(x)=x\cdot c^r \quad (\text{均在}G\text{中运算})$$

随同这个加密消息发送的还有“头部” $b^r\in G$。注意加密者只需知道 b,c,G，并选择随机数 $r\in \boldsymbol{Z}$，但不知道离散对数 $l\in \boldsymbol{Z}$。

合法的**解密步骤**需要知道离散对数 l 的信息，但不需要随机数 r。首先由“头部” $b^r\in G$，解密者计算

$$(b^r)^l=(b^l)^r=c^r$$

然后给上面等式两边同时乘以 c^r 在 G 中的乘法逆元 $(c^r)^{-1}$，即可恢复明文消息：

$$D_{b,c,p,r,l}(y)=(c^r)^{-1}\cdot y=(c^r)^{-1}\cdot c^r\cdot x=x\in G$$

备注 群 G 中离散对数密码的安全性当然依赖于 G 中离散对数计算的困难性，即给定 b 和 $c\in G$，并确保**存在**某个 $l\in \boldsymbol{Z}$，使 $b^l=c$，要想**求得** l 是非常困难的。

注意，将群中的运算记为“乘法”或者“加法”仅仅是一个记号。如果需要更加正式的记法，那么群的运算记为“$*$”，则“指数运算”就是反复做“$*$”运算，也就是

$$b^l=\underbrace{b*b*\cdots*b}_{l}$$

特别地，如果将群运算记为“加法”，则“指数运算”实际上就是乘法：

$$b^l=\underbrace{b+b+\cdots+b}_{l}=l\cdot b$$

因此，当 G 的运算为模整数 n 的**加法**时，很容易计算离散对数。原因就是此时的指数实

际上是模整数 n 的乘法。因此从方程

$$b^l = c \text{(此处的群运算为模整数 } n \text{ 的加法)}$$

求解 l 就是求解 $l \cdot b = c$，这很容易得到 $l = b^{-1} \cdot c \bmod n$。

当然，在不同的群上，寻找模整数 n 的乘法逆元也可能是困难的，但在上面的例子中可用**欧几里得算法**求得 b 的乘法逆元 b^{-1}。所以**并不是所有的**离散对数都是难以计算的。但相比而言，目前所有可用的证明似乎表明在群 $\mathbf{Z}/p^{\times}$ 中计算离散对数是困难的。尽管对 $\mathbf{Z}/p^{\times}$ 加法群求离散对数是容易的，但对乘法群 $\mathbf{Z}/p^{\times}$ 求离散对数却是困难的。目前也没有已知的定理**证明**在 $\mathbf{Z}/p^{\times}$ 上求解离散对数是困难的，就如同没有已知的定理证明大整数分解是困难的一样。但是，所有可用的推测性证据表明，对于大素数 p，在 $\mathbf{Z}/p^{\times}$ 上计算离散对数几乎与分解大整数 n 一样困难。

但是，就如同我们已经看到某些“还不算太差”的算法可以比穷举试除法更好地分解大数一样，我们也有一些“还不算太差”的算法来计算离散对数。这些算法的速度还不足以称它们为“快速算法”。

28.3 椭圆曲线上的运算

为了叙述简便，我们以 $\mathbf{Z}/p$（$p \neq 2,3$）上的椭圆曲线为例。给定 $\mathbf{Z}/p$ 的元素 a 和 b，使得方程 $x^3 + ax + b = 0$ 在 $\mathbf{Z}/p$ 中没有重根。（我们已经知道如何去验证没有重根这个条件，只要计算 $x^3 + ax + b$ 与其“形式”导数 $3x^2 + a$ 的最大公因子即可。）这样，一条椭圆曲线就是一些点（实际上为 $\mathbf{Z}/p$ 的元素对）的集合：

$$E = \{(x, y) \in \mathbf{Z}/p \times \mathbf{Z}/p : y^2 = x^3 + ax + b\}$$

对于 $\mathbf{Z}/2$、$\mathbf{Z}/3$ 以及**任何有限域** $\mathbf{F}_q = GF(q)$，可按相同的方法构造椭圆曲线，除了那些必须考虑的最一般的形式：

$$y^2 + (c + x)y = x^3 + ax^2 + b \text{（特征为 2）}$$

$$y^2 = x^3 + ax^2 + bx + c \text{（特征为 3）}$$

当所考虑的域（如同上面的 $\mathbf{Z}/p$）为 $\mathbf{R}$ 和 $\mathbf{C}$ 时，这样的构造也是有意义的。此时，我们还可以描绘出有意义的曲线图像。

备注 还不太明确是否存在一种合理的方法把这些点做成一个群，或者这些点上是否存在某种自然的群结构。事实上，的确存在这样一种自然的群结构。

现在我们来定义**椭圆曲线上的群运算**。考虑椭圆曲线 $E = \{y^2 = x^3 + ax + b\}$，稍后我们将用到一个神秘的**无穷远点**，一般记为 $\mathbf{O}$。这里暂且不解释如何使其合理化，群的运算记为“+”。

- 对曲线上两个不同的点 $\mathbf{P}$ 和 $\mathbf{Q}$，这两个点均不为 $\mathbf{O}$ 且有不同的 $x-$ 坐标。令 L 是连接 $\mathbf{P}$ 和 $\mathbf{Q}$ 的直线，我们可以验证 L 与 E 恰有 3 个交点，其中两个就是 $\mathbf{P}$ 和 $\mathbf{Q}$，令第三个点为 (x, y)，则定义 E 上的**加法**为

$$\mathbf{P} + \mathbf{Q} = (x, -y)$$

- 如果点 $\mathbf{P}$ 和 $\mathbf{Q}$ 有相同的 $x-$ 坐标，但 $\mathbf{P} \neq \mathbf{Q}$，则定义

$$\mathbf{P} + \mathbf{Q} = \mathbf{O}$$

- 如果 $\mathbf{P} = \mathbf{Q} = (x_0, y_0)$ 且 $y_0 \neq 0$，令 L 是在 $\mathbf{P}$ 点与 E **相切的**直线，并令 (x, y) 是 L 与 E 的第三个交点，则定义

$$\mathbf{P}+\mathbf{P}=(x,-y)$$

- 如果$\mathbf{P}=(x,0)$是E上的点，则定义$\mathbf{P}+\mathbf{P}=\mathbf{O}$；
- 对E上的任意点$\mathbf{P}$，$\mathbf{P}+\mathbf{O}=\mathbf{O}+\mathbf{P}=\mathbf{P}$；
- 对E上的任意点$\mathbf{P}=(x,y)$（不是$\mathbf{O}$），则$-\mathbf{P}=(x,-y)$，且$-\mathbf{O}=\mathbf{O}$。

那么此时我们会产生如下疑问：

- 这个运算为什么是正确的？
- 该运算是如何计算的？
- 它为什么是满足结合律的（例如……）？
- 这种思想又是如何获得的？

非常不幸的是，这些问题没有一个是很容易解答的。我们提出这些问题的目的仅仅是声明这些问题对进一步学习椭圆曲线是非重要的。

备注 如果我们用普通的“描点作图法”绘制出曲线的图，尽管看上去几何语言是有意义的，但“切线”的含义一点都不明确，比如我们所考虑的域为有限域。关键的问题是“直线”的概念必须完全用代数的语言解释，而不是依赖于“几何上的直觉”。在后面的介绍中，我们还需要进一步解释“直线”为什么与这样一个三次曲线有三个交点？是不是一条直线与一个n次曲线就一定会有n个交点？

下面我们来推导出椭圆曲线上两点之**和的公式**。设椭圆曲线E是满足$y^2=x^3+ax+b$的点(x,y)的集合，给定两点$\mathbf{P}_1=(x_1,y_1)$和$\mathbf{P}_2=(x_2,y_2)$。假定$y_1\neq y_2$，计算两点之和如下。

通过两点(x_1,y_1)和(x_2,y_2)的直线L就是满足下式的点(x,y)的集合

$$\frac{y-y_1}{x-x_1}=\frac{y_2-y_1}{x_2-x_1}$$

对上式变形即得

$$y=y_1+\frac{y_2-y_1}{x_2-x_1}(x-x_1)$$

进一步变形为

$$y=\frac{y_2-y_1}{x_2-x_1}x+\frac{x_2y_1-y_2x_1}{x_2-x_1}$$

为简便起见，我们将其记为$y=Ax+B$，然后将此式代入三次方程式即得：

$$(Ax+B)^2=x^3+ax+b$$

重新整理这个关于x的三次方程即得：

$$x^3-A^2x^2+(-2AB+a)x+(b-B^2)=0$$

对任意首项系数为1的三次方程，二次项的系数即为方程根的和的负数，这一点容易由下式得到验证：

$$(x-\alpha)(x-\beta)(x-\gamma)=x^3-(\alpha+\beta+\gamma)x^2+(\alpha\beta+\beta\gamma+\gamma\alpha)x-\alpha\beta\gamma$$

因此，如果三次曲线上第三个点的横坐标为x_{sum}，则x^2项的系数的负值就等于未知的横坐标x_{sum}加上x_1和x_2，即

$$A^2=x_1+x_2+x_{\text{sum}}$$

暂且不考虑A是什么，我们首先得出x_{sum}的表示式：

$$x_{\text{sum}} = A^2 - x_1 - x_2 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2$$

将 x_{sum} 代入前面的直线方程 $y = Ax + B$，则有

$$Ax_{\text{sum}} + B = \frac{y_2 - y_1}{x_2 - x_1} x_{\text{sum}} + \frac{x_2 y_1 - y_2 x_1}{x_2 - x_1}$$

由定义，取负即得和的纵坐标，即

$$y_{\text{sum}} = -\frac{y_2 - y_1}{x_2 - x_1} x_{\text{sum}} - \frac{x_2 y_1 - y_2 x_1}{x_2 - x_1}$$

在实际应用中，还需要用到公式 $2 \cdot \mathbf{P} = \mathbf{P} + \mathbf{P}$。

这就是说，给定椭圆曲线 E 上的点 $\mathbf{P} = (x_1, y_1)$，令 L 是 E 在 $\mathbf{P} = (x_1, y_1)$ 点的切线，找出这个直线与曲线 E 的另一个交点，并将其纵坐标取负，即得 $(x_{\text{sum}}, y_{\text{sum}})$。曲线的斜率为 $\frac{dy}{dx}$，我们可以用导数来计算：

$$2y \cdot \frac{dy}{dx} = 3x^2 + a$$

从而

$$\frac{dy}{dx} = \frac{3x^2 + a}{2y}$$

代入 $\mathbf{P}$ 点的坐标值 (x_1, y_1)，即得

$$\frac{dy}{dx}\Big|(x_1, y_1) = \frac{3{x_1}^2 + a}{2y_1}$$

因此切线方程则为

$$y - y_1 = \frac{3{x_1}^2 + a}{2y_1}(x - x_1)$$

或者

$$y = y_1 + \frac{3{x_1}^2 + a}{2y_1}(x - x_1)$$

同样将 y 的值代入椭圆曲线方程 $y^2 = x^3 + ax + b$，重新整理可得到一个关于 x 的三次方程：

$$\left(y_1 + \frac{3{x_1}^2 + a}{2y_1}(x - x_1)\right)^2 = x^3 + ax + b$$

或者

$$x^3 - \left(\frac{3x_1^2 + a}{2y_1}\right)^2 x^2 + (\text{线性项和常数项}) = 0$$

由于三次方程的根的和为二次项系数的负值，并且因为切线给出了**两个**交点，所以两个根满足

$$x_{\text{new}} + x_1 + x_1 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2$$

由此即得

$$x_{\text{new}} = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1$$

将 x_{new} 代入前面的直线方程，我们得到

$$y_1 + \frac{3x_1^2 + a}{2y_1}(x_{\text{new}} - x_1)$$

由定义取**负值**即可得到和的纵坐标：

$$y_{\text{sum}} = -y_1 - \frac{3x_1^2 + a}{2y_1}(x_{\text{new}} - x_1)$$

这就是椭圆曲线上一个点 $\mathbf{P} = (x_1, y_1)$ 的**两倍**的坐标公式。注意到这个“异样的”群运算可由加法、乘法和求逆来完成。

为什么这种运算使椭圆曲线成为一个群呢？这一问题的关键部分就是**为什么它是可结合的**。交换律容易由公式验证。对这个普通的问题却有三种形式的答案，并且没有一个是简单的。

- 由**复变函数论**我们可以得知，复数域上的椭圆曲线有一个群法则，因为它是复数域的**商群**。这也仅仅指出当所讨论的域为其他域时，应当有一个群法则。学习数学的人在关注有限域上的椭圆曲线之前，最好多学一些复变函数论以掌握复数域上的椭圆曲线。
- 在**射影几何**理论中，存在一个特别难以理解但又似乎是基本的结合律证明。按照现代的观点，这是一个过于专业化且不是太好的试图理解椭圆曲线上群法则的方法。但是，依据这些方法的创设情况，可以给出一个证明。
- 从技术上看是最困难的方法，但从长远看它却给出了最清晰的观点，这样的方法在现代代数曲线理论中有其理论根源，特别是在比较重要且微妙的**黎曼-罗赫定理**中。这一观点真正给出了椭圆曲线是一个群的证明，并且给出了其他类型曲线所涉及的思想方法。

下面我们来做一些椭圆曲线上的简单计算。令 E 为域 $\mathbf{F}_{17}$ 上的椭圆曲线 $y^2 = x^3 + x$，通过反复试验，我们找到曲线上的一个点 $(1,6)$，因为 $6^2 = 1^3 + 1 \bmod 17$。运用椭圆曲线上的群法则来计算 $(1,6) + (1,6) = 2 \cdot (1,6)$：两点之和的横坐标

$$x_{\text{sum}} = \left(\frac{3 \cdot 1^2 + 1}{2 \cdot 6}\right)^2 - 2 \cdot 1 = \left(\frac{1}{3}\right)^2 - 2 = 6^2 - 2 = 0 \bmod 17$$

而纵坐标

$$y_{\text{sum}} = -6 - \left(\frac{3 \cdot 1^2 + 1}{2 \cdot 6}\right)(0 - 1) = 0 \bmod 17$$

因此，

$$(1,6) + (1,6) = 2 \cdot (1,6) = (0,0)$$

备注 这里的运算并**不是**向量的加法或者标量的乘法，尽管看上去有点像。

继续在椭圆曲线 $y^2 = x^3 + x \bmod 17$ 上计算

$$3 \cdot (1,6) = (1,6) + 2 \cdot (1,6) = (1,6) + (0,0)$$

仍然运用前面的公式，

$$x_{\text{sum}} = \left(\frac{0-6}{0-1}\right)^2 - 1 - 0 = 1 \bmod 17$$

$$y_{\text{sum}} = -\left(\frac{0-6}{0-1}\right)^2 \cdot 1 - \frac{0\cdot 6 - 0\cdot 1}{0-1} = 11 \bmod 17$$

继续：如果我们再计算 $(1,6)+(1,11)$，则由群法则的一般定义，得到了**无穷远点 O**，这表明 $\mathbf{P}=(1,6)$ 在群 E 上的**阶**为 4。

再次通过试验法，在由 $\{(x,y): y^2 = x^3 + x \bmod 17\}$ 给出的椭圆曲线上，如果包含无穷远点 **O**，我们总共找到了 16 个点：

$$(0,0)$$
$$(1,\pm 6)$$
$$(3,\pm 8)$$
$$(4,0)$$
$$(6,\pm 1)$$
$$(11,\pm 4)$$
$$(13,0)$$
$$(14,\pm 2)$$
$$(16,\pm 7)$$

由拉格朗日定理，E 中元素的可能阶为 1、2、4、8、16。阶为 1 的元素只有元素 **O**，有三个点的纵坐标为 0，即 (0,0)、(4,0)、(13,0)，由定义它们是阶为 2 的元素。

通过试验我们还发现

$$2\cdot(1,\pm 6) = (0,0)$$
$$2\cdot(3,\pm 8) = (13,0)$$
$$2\cdot(6,\pm 1) = (13,0)$$
$$2\cdot(11,\pm 4) = (4,0)$$
$$2\cdot(6,\pm 1) = (13,0)$$
$$2\cdot(14,\pm 2) = (4,0)$$
$$2\cdot(16,\pm 7) = (0,0)$$

由此，这些点的阶为 4，故 E 上没有 8 阶的元素，也没有 16 阶的元素。

28.4 无穷远点

为了使**无穷远点**更为合理，我们可以放大平面，使它更适合于多项式方程和代数曲线的研究。具体如下，设 k 为任意域，包括 **C** 和 **R** 以及有限域 $\mathbf{F}_q$。设 k^n 为由 k 的有序 n 元组构成的空间，这是一个**仿射 n 空间**。

令 $\Omega = \{(x,y,z) \in k^3 : x,y,z \text{不全为0}\}$，即 Ω 为 k^3 去掉 (0,0,0)。在 Ω 上定义一个**等价关系** $\backsim$，$(x,y,z) \backsim (x',y',z')$，如果对某个 $\lambda \in k^{\times}$，有 $(x,y,z) = (\lambda x', \lambda y', \lambda z')$。那么 k 上的**射影平面** $\mathbf{P}^2$ 定义为 Ω 上等价类的集合，

$$\mathbf{P}^2 = \Omega / \backsim$$

坐标 (x,y,z) 就是 $\mathbf{P}^2$ 中点的**齐次坐标**。

比如 $k = \mathbf{Q}$，使用 $\mathbf{P}^2$ 上的齐次坐标，

$$(1,2,3) \backsim (4,8,12) \backsim (-1,-2,-3)$$
$$(0,2,0) \backsim (0,1,0) \backsim (0,-1,0)$$

通过映射 $(x,y) \to (x,y,1)$，**仿射平面** k^2 可以嵌入到平面 $\mathbf{P}^2$ 上。注意到 $(x,y,1) \backsim (x',y',1)$，当且仅当 $(x,y)=(x',y')$。

因此，比如仿射平面上的点 $(3,4)$，在射影平面 $\mathbf{P}^2$ 上用齐次坐标表示则为 $(3,4,1)$。

无穷远直线就是 $\mathbf{P}^2$ 中点 (x,y,z) 且 $z \neq 0$ 的集合。注意到仿射平面 k^2 的嵌入平面上不存在这样一个点，因此不论它们是什么，它们总是存在于普通的平面之外。

射影空间上的曲线。我们介绍 $\mathbf{P}^2$ 的动机就是使初等几何中的陈述变得平滑且对称，比如在普通的平面 $\mathbf{R}^2$ 上，任何两条（不同的）直线恰好相交于一点，除非它们平行。这就有一点不对称，我们可以修正它。我们可以说两条平行直线相交于无穷远点，当然这没有什么实质内容，但我们可以用 $\mathbf{P}^2$ 理解它的含义。

由直线 L 的方程

$$ax+by+c=0\ (a,b \text{ 不能全为 } 0)$$

我们可以构造**相伴的齐次方程**

$$ax+by+cz=0$$

该方程具有这样的性质，如果 (x,y,z) 满足方程，则对任何 $\lambda \in k^{\times}$，$(\lambda x,\lambda y,\lambda z)$ 亦满足方程。即齐次化的方程定义了**射影空间 P** 上的一条**曲线** $\widetilde{L}$，而且我们希望它在 k^2 嵌入到平面 $\mathbf{P}^2$ 的作用下，原来的直线 L 被映射为 $\widetilde{L}$ 的一个子集。

那么在 L 的扩展形式 $\widetilde{L}$ 上，**无穷远点**是什么呢？这需要我们来考查 $ax+by+cz=0$ 且 $z=0$ 的解：也就是要考查 $ax+by=0$。因为 a,b 不能全为 0，不失一般性，假设 $b \neq 0$，则得

$$y=(-a/b)x$$

由此我们得到 $\widetilde{L}$ 上的点

$$(x,-ax/b,0) \backsim (1,-a/b,0)$$

即这是对同一个点的不同齐次坐标表示：在给定的直线上存在一个无穷远点。

所以我们有如下平滑对称的论断：

定理 射影平面 $\mathbf{P}^2$ 上任何两条（不同）直线恰好相交于一点。

证明 设 $\mathbf{P}^2$ 上两条直线为（按照齐次坐标表示）

$$ax+by+cz=0$$
$$a'x+b'y+c'z=0$$

假设这两条直线是不相同的，这一点可以验证，实际上需要验证 (a,b,c) 不是 (a',b',c') 的标量乘（等价地，(a',b',c') 也不是 (a,b,c) 的标量乘）。我们必须求解方程组以得到交点的坐标 (x,y,z)。

由基本的线性代数知识，我们需要寻找满足下式的向量 (x,y,z)：

$$(x,y,z)\cdot(a,b,c)=0$$
$$(x,y,z)\cdot(a',b',c')=0$$

这里的运算“$\cdot$”为普通的**点积**。由实数域上的线性代数的知识，我们希望 (a,b,c) 和 (a',b',c') 的**叉积**是一个解，即

$$(x,y,z)=(bc'-b'c,-ac'+a'c,ab'-a'b)$$

实际上，我们发现

$$a(bc'-b'c)+b(-ac'+a'c)+c(ab'-a'b)=0$$
$$a'(bc'-b'c)+b'(-ac'+a'c)+c'(ab'-a'b)=0$$

注意到 (a,b,c) 和 (a',b',c') 之间不存在标量乘关系，因此这使得

$$(x,y,z)=(bc'-b'c,-ac'+a'c,ab'-ba')$$

不可能为 (0,0,0)。

进一步的考查还表明，所有解的集合恰好是这个解的标量乘的集合。因此，考虑到 $\mathbf{P}^2$ 上所定义的等价关系，只有惟一的解。 ♣

28.5 射影椭圆曲线

由方程 $y^2=x^3+ax+b$ 所确定的椭圆曲线 E 有一个**齐次形式**，

$$y^2z=x^3+axz^2+bz^3$$

我们给每一项插入关于 z 的因子，使每一项的次数都为 3。这样做的结果是，如果 (x,y,z) 满足方程，则对任意 $\lambda\in k^{\times}$，$(\lambda x,\lambda y,\lambda z)$ 亦满足方程。

我们感兴趣的一个问题就是**无穷远点**是否在扩展的椭圆曲线上：令 $z=0$，即求解如下方程

$$0\cdot y^2=x^3+ax\cdot 0^2+b\cdot 0^3 \qquad \text{或者 } x^3=0$$

因此惟一的无穷远点由齐次坐标 (0,1,0) 给出。这就是 **O**，椭圆曲线上群法则中的**单位元**。

这样，我们得到了射影坐标表示的椭圆曲线上的群法则。考虑由

$$y^2z=x^3+axz^2+bz^3$$

的齐次坐标给出的椭圆曲线 E，给定扩展的椭圆曲线上的两个不同点 (x,y,z) 和 (x',y',z')，令 (a,b,c) 为通过这两个点的直线 $\tilde{L}$ 的系数，这些系数可由下面两个式子确定：

$$ax+by+cz=0$$
$$ax'+by'+cz'=0$$

因为 (x,y,z) 和 (x',y',z') 是 $\mathbf{P}^2$ 上不同的点，所以这个方程组的所有解 (a,b,c) 之间存在标量乘关系，这样就得到了 $\mathbf{P}^2$ 上的一条直线 $\tilde{L}$。

$\tilde{L}$ 与椭圆曲线的交点为 (x_s,y_s,z_s)（计算方法同前），因此由 $y-$ 坐标取负值即得两点之和的定义：

$$(x,y,z)+(x',y',z')=(x_s,-y_s,z_s)$$

2·**P** 的一般公式也可以用类似的方法得到。

第29章 有限域的更多知识

迄今我们所讨论的有限域以及系数在这些域上的多项式，如果只着眼于计算的目的，可以说是足够了，但要做出进一步的解释就有点欠缺。从长远的观点和目的出发，我们还需要很多这方面的知识。本章我们将从略微大一点且更加抽象的观点来考查有限域的构造。这不是绝对必要的，但它可以帮助我们理解有限域存在的模式以及本质，而不是对这些概念和理论仅有一个无序的认识。

这些理论化的进展也仅仅是我们的最低要求。本章所讨论的域理论专门适用于有限域的情况，而且这里的讨论也是非常有限的。另一方面，要给出域理论的一般介绍则可能需要更多的时间和篇幅。

29.1 交换环上的理想

交换环上**理想**的概念是数的概念的一种推广。实际上，最初有一个非常接近的概念，称为**理想数**，它就是扩展了数的通常概念。这个短语从现在开始简称为“理想”。

设 R 是单位元为1的交换环。R 的一个**理想**是 R 的一个子集 I，使得

- 对所有的 $r\in R$ 和 $i\in I$，有 $r\cdot i\in I$。（对环中元素的乘法封闭）
- 对所有的 $i,j\in I$，有 $i+j\in I$。（对加法封闭）
- 对所有的 $i\in I$，有 $-i\in I$。（对加法逆封闭）
- $0\in R$。

第二、第三和第四个条件可以概述为在加法运算下，要求 I 是 R 一个子群。第一个条件似乎有一点特别。首先，这个要求比起 I 是 R 一个子环的要求要强一点，因为这里要求 I 对 R 中的元素的乘法是封闭的，并不仅仅局限于 I 自身的元素。

例子 基本的例子如下，在环 $\mathbf{Z}$ 中，对于任意给定的整数 n，包含所有 n 的倍数的集合 $n\cdot\mathbf{Z}$ 是一个理想。实际上，如果 $x=mn$ 为一个 n 的倍数，并且如果 $r\in\mathbf{Z}$，则 $r\cdot x=r(mn)=(rm)n$ 仍然为一个 n 的倍数。同样地，$n\cdot\mathbf{Z}$ 包含0，它对求和封闭且对加法逆封闭。

例子 设 $R=k[x]$ 为变元为 x 且系数在域 k 中的多项式环。给定一个多项式 $P(x)$，令 $I\subset R$ 为所有 $P(x)$ 倍式 $M(x)\cdot P(x)$ 的集合。I 是一个理想，其验证步骤形式上等同于前面的例子。

例子 不同于前面两个例子，这里给出一个抽象的例子。设 R 是任意单位元为1的交换环，且给定 $n\in R$。则所有包含 n 的倍数的集合 $I=n\cdot R=\{rn:r\in R\}$ 是一个理想，称为由 n **生成的主理想**。同样可以证明它是理想，这样的理想称为**主理想**。

例子 在任意环上，**平凡理想**就是指 $I=\{0\}$ 和 $I=R$。为了和数学中典型的用法一致，我们称一个理想是**真**理想，如果它既不是理想 $\{0\}$，也不是整个环 R。

下面的命题是一个重要的基本原理。

命题 设 I 是单位元为1的交换环 R 的理想，如果 I 包含任意的元素 $u\in R^{\times}$，则 $I=R$。

证明 假设 I 包含 $u\in R^{\times}$，u 是一个单位即意味着存在一个 u 的乘法逆元 u^{-1}，对于任意的 $r\in R$，

$$r=r\cdot 1=r\cdot(u^{-1}\cdot u)=(r\cdot u^{-1})\cdot u$$

这也就是说 r 是 u 的倍数。因为 I 是一个理想，它必定包含 u 的每个倍数，所以 I 包含 r。因为这一结论对每个元素 $r\in R$ 都成立，所以 $R=I$。 ♣

推论 设 I 是多项式环 $k[x]$ 的一个理想，k 为一个域。如果 I 包含任何非零的“常数”多项式，则 $I=k[x]$。

证明 如果我们能够验证非零常数多项式是单位（即有一个乘法逆元），则可由前面的命题直接得到结论。实际上，对 $a\in k$ 且 $a\neq 0$，因为 k 是一个域，存在 $a^{-1}\in k\subset k[x]$。因此 a 在多项式环 $k[x]$ 中是可逆的。 ♣

我们可以利用以前用于陪集的记号，这样可使理想的表示更加简洁。对于环 R 的两个子集 X,Y，记

$$X+Y=\{x+y:x\in X,y\in Y\}$$

$$X\cdot Y=XY=\{\text{有限和}\sum ix_iy_i:x_i\in X,y_i\in Y\}$$

注意在环论中，记号 $X\cdot Y$ 的含义与群论中的含义不同。我们可以说*交换环 R 的理想 I 是一个加法子群，且使得 $RI\subset I$*。

命题 $\mathbf{Z}$ 的每个理想都是*主理想*，即它具有 $I=n\cdot\mathbf{Z}$ 的形式。特别地，使得这一命题成立的整数 n 是 I 的最小正整数，除非 $I=\{0\}$。$I=\{0\}$ 时，$n=0$。

证明 如果 $I=\{0\}$，则当然有 $I=\mathbf{Z}\cdot 0$，结论自然成立。假设 I 是非零的，因为 I 对取加法逆是封闭的，如果 I 包含 $x<0$，则它也包含 $-x>0$。所以一个非平凡的理想 I 包含一些正的元素。设 n 是 I 的最小元素，$x\in I$，则利用除法算法可以得到 $q,r\in\mathbf{Z}$，且 $0\leqslant r<n$，满足

$$x=q\cdot n+r$$

qn 仍然属于 I，那么 $-qn$ 也属于 I。因为 $r=x-qn$，我们断定 $r\in I$。因为 n 是 I 中的最小正元素，所以必有 $r=0$。因此 $x=qn\in n\cdot\mathbf{Z}$，命题得证。 ♣

命题 设 k 为一个域，令 $R=k[x]$ 是变元为 x 且系数在域 k 中的多项式环，则 R 的每个理想 I 都是主理想，也就是对某个多项式 P，它具有 $I=k[x]\cdot P(x)$ 的形式。特别地，$P(x)$ 是 I 中次数最小的首一多项式，除非 $I=\{0\}$。$I=\{0\}$ 时，$P(x)=0$。

证明 如果 $I=\{0\}$，则当然有 $I=k[x]\cdot 0$，结论自然成立。假设 I 是非零的，令 $Q(x)=a_nx^n+\ldots+a_0$ 是 I 中的多项式，且 $a_n\neq 0$。因为 k 为一个域，所以 a_n 有一个逆元素 a_n^{-1}。又因为 I 是一个理想，则多项式

$$P(x)=a_n^{-1}\cdot Q(x)=x^n+a_n^{-1}a_{n-1}x^{n-1}+\ldots+a_n^{-1}a_0$$

也属于 I。也就是在 I 中确实存在一个次数最小的首一多项式。令 $x\in I$，利用除法算法我们得到 $Q,R\in k[x]$，$\deg R<\deg P$ 以及

$$x=Q\cdot P+R$$

$Q\cdot P$ 属于 I，当然有 $-Q\cdot P\in I$。因为 $R=x-Q\cdot P$，我们断定 $R\in I$。因为 P 是 I 中次数最小且首一的多项式，则必有 $R=0$。因此，$x=Q\cdot P\in n\cdot k[x]$。

命题得证。 ♣

备注 这两个命题的证明经过抽象后可用来证明欧几里得环中的每个理想都是主理想。

例子 设 R 是单位元为1的交换环，给定两个元素 $x,y\in R$，则

$$I=R\cdot x+R\cdot y=\{rx+sy:r,s\in R\}$$

是 R 的一个理想。验证步骤如下，首先，$0=0\cdot x+0\cdot y$，所以 $0\in I$；其次，由于 $-(rx+sy)=(-r)x+(-s)y$，所以 I 对取逆是封闭的；第三，对 I 中的两个元素 $rx+sy$ 和 $r'x+s'y$（$r,r',s,s'\in R$）我们有

$$(rx+sy)+(r'x+s'y)=(r+r')x+(s+s')y$$

所以 I 对加法是封闭的；最后，对于 $rx+sy\in I$（$r,s\in R$）和 $r'\in R$，我们有

$$r'\cdot(rx+sy)=(r'r)x+(r's)y$$

所以 $R\cdot I\subset I$。因为这种类型的 I 的确是一个理想。两个元素 x,y 是 I 的生成元。

例子 同一面的例子类似，给定交换环 R 的元素，我们可以构造如下形式的理想：$I=R\cdot x_1+\cdots+R\cdot x_n$。

例子 利用我们已知的较小的理想，可以按照如下方式构造新的更大的理想。设 I 是交换环 R 的一个理想，$x\in R$，则令

$$J=R\cdot x+I=\{rx+i:r\in R,i\in I\}$$

我们来验证 J 是一个理想。首先，$0=0\cdot x+0$，所以 $0\in J$；其次，$-(rx+i)=(-r)x+(-i)$，所以 J 对取逆是封闭的；第三，对于 J 中的两个元素 $rx+i$ 和 $r'x+i'$（$r,r'\in R$ 且 $i,i'\in I$），我们有

$$(rx+i)+(r'x+i')=(r+r')x+(i+i')$$

所以 J 对加法是封闭的；最后，对 $rx+i\in J$，$r\in R$，$i\in I$ 以及 $r'\in R$，

$$r'\cdot(rx+i)=(r'r)x+(r'i)$$

所以 $R\cdot J\subset J$。因此这种类型的集合 J 的确是一个理想。

备注 在我们已经知道每个理想都是主理想的情况，如 $\mathbf{Z}$，前面的构造方法不会产生更加一般的理想。

备注 在某些环中，并非每个理想都是主理想。也就是有一些理想不能被表示为 $R\cdot x$。这种情况的例子如下。设

$$R=\{a+b\sqrt{-5}:a,b\in\mathbf{Z}\}$$

不难验证这是一个环。令 $I=\{x\cdot 2+y\cdot(1+\sqrt{-5}):x,y\in R\}$，稍加注意我们就会发现这个理想不是主理想。这一现象与在该环中不能进行惟一因子分解关系十分密切。比如，我们明显可以得出两个不同的分解：

$$2\cdot 3=6=(1+\sqrt{-5})\cdot(1-\sqrt{-5})$$

（四个数 $2,3,1+\sqrt{-5},1-\sqrt{-5}$ 都是“素数”，但它们在环 R 中不能更进一步地分解。）这些现象的产生，促进了历史上代数数论的发展。

备注 在非交换的环 R 上，存在三种不同的理想。一个**左理想** I 是一个加法子群，使得 $RI\subset I$，一个**右理想**是一个加法子群，使得 $IR\subset I$，而一个**双边理想**也是一个加法子群，使得 $RI+IR\subset I$。通常我们仅关注交换环上的理想，所以经常忽略这些因素。

习题

29.1.01 设 N 是一个整数，详细证明 $N\cdot\mathbf{Z}$ 是 $\mathbf{Z}$ 的一个理想。

29.1.02 找出 $\mathbf{Z}/6$ 中的所有理想。

29.1.03 找出 $\mathbf{Z}/10$ 中的所有理想。

29.1.04 找出 $\mathbf{Z}/5$ 中的所有理想。

29.1.05 设 I,J 是环 R 的两个理想，证明 $I\cap J$ 也是 R 的一个理想。

29.1.06 设 I,J 是环 R 的两个理想，令 $I+J=\{i+j:i\in I,j\in J\}$，证明 $I+J$ 是一个理想。

29.1.07(*) 证明有两个变元 x,y 且系数在有限域 $\mathbf{F}_2$ 上的多项式环 $\mathbf{F}_2[x,y]$ 不是一个主理想整环。

29.2 环同态

与群同态十分类似，环同态是一个环到另一个环并且保持环的结构的映射。准确地说，一个**环同态** $f:R\to S$ 是一个环 R 到另一个环 S 的映射，并且使得对所有的 $r,r'\in R$，有

$$f(r+r')=f(r)+f(r')$$
$$f(rr')=f(r)f(r')$$

也就是说 f 保持了环的加法和乘法运算。一个环同态如果是双射，那么它就是一个**同构**。在整个环理论的介绍中，两个同构的环在结构上可以被视为“相同的”。

正如在群和群同态的讨论中一样，我们也不希望用不同的记号来表示两个不同的环 R 和 S 定义中的加法和乘法。因此，同态 f 将 R 中的*加法*转换为 S 中的*加法*，也同样将 R 中的乘法转换为 S 中的乘法。

非常类似于群中的情况，环同态 $f:R\to S$ 的**核**为 $\ker f=\{r\in R:f(r)=0\}$，这里的 0 应当是 S 中的加法单位元。

例子 最基本的环同态的例子为 $f:\mathbf{Z}\to\mathbf{Z}/n$，其定义为 $f(x)=x-\text{mod}-n$。f 只所以是环同态，原因是我们有如下两个等式：

$$(x-\text{mod}-n)+(y-\text{mod}-n)=(x+y)-\text{mod}-n$$
$$(x-\text{mod}-n)\cdot(y-\text{mod}-n)=(x\cdot y)-\text{mod}-n$$

尽管这种叫法稍微有一点误导，但我们仍称这个同态为**模 m 约简同态**。

命题 任意环同态 $f:R\to S$ 的核是 R 的一个理想。

证明 设 x 是环同态 $f:R\to S$ 的核中的一个元素，$r\in R$，则

$$f(rx)=f(r)f(x)=f(r)\cdot 0=0$$

因为我们已经证明在环中任意元素乘以 0 均得到 0。因此 rx 就是 f 的核中的元素。对于核中的两个元素 x,y，我们有

$$f(x+y)=f(x)+f(y)=0+0=0$$

也就是说 $x+y$ 也属于核。又因为 $f(0)=0$，所以 0 属于核。对于核中的元素 x，由于 $f(-x)=-f(x)=-0=0$，所以 $-x$ 也在核中。因此，环同态 $f:R\to S$ 的核是 R 的一个理想。♣

例子 有一些在实际应用中非常重要的同态，称为**赋值同态**或**代替同态**，具体描述如下。设 R 为一个交换环，$R[x]$ 为有一个变元 x 且系数在 R 上的多项式环。给定 $r_0\in R$，我们将给多项式赋值 r_0，或者等价地说，用 r_0 来代替多项式中的 x。这么做的含义是什么呢？这也就是将多项式

$$P(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_2x^2+a_1x+a_0$$

映射到

$$P(r_0)=a_n{r_0}^n+a_{n-1}{r_0}^{n-1}+\cdots+a_2{r_0}^2+a_1r_0+a_0$$

我们用 e_{r_0} 来表示这个映射，它就是一个**赋值映射**。

- 赋值映射 $e_{r_0}:R[x]\to R$ 是一个由多项式环 $R[x]$ 到环 R 的同态。

备注 在证明这个结论之前，注意到我们的经验会让我们希望这个映射真正是一个环同态：实际上，我们知道给两个多项式的积或和赋值，我们可以单独给它们先赋值然后再相乘/相加，或者先相乘/相加然后再赋值。这个赋值的确是一个环同态。

证明 这里需要使用有效的记号。设 $P(x)=\sum_{0\le i\le m}a_ix^i$，$Q(x)=\sum_{0\le i\le n}b_ix^i$ 为两个系数在环交换 R 中的多项式。首先我们来证明在 $r_0\in R$ 的赋值映射 e_{r_0} 保持加法运算：

$$e_{r_0}(P+Q)=e_{r_0}\left(\sum_j(a_j+b_j)x^j\right)=\sum_j(a_j+b_j)r_0^j$$

$$=\sum_j a_jr_0^j+\sum_j b_jr_0^j=e_{r_0}(P)+e_{r_0}(Q)$$

这里，对于指标超出了系数所定义的范围，我们令 $a_j=0$ 且 $b_j=0$。这样我们就证明了赋值映射是保持求和运算的。对于乘法，我们有：

$$e_{r_0}(P\cdot Q)=e_{r_0}\left(\sum_{i,j}(a_i\cdot b_j)x^{i+j}\right)=\sum_{i,j}(a_i\cdot b_j)r_0^{i+j}=e_{r_0}(P)\cdot e_{r_0}(Q)$$

这就证明了赋值映射保持乘法。因此赋值映射的确是一个环同态。 ♣

命题 设 $f:R\to S$ 是一个环同态，$0_R,0_S$ 分别是 R 和 S 的加法单位元，则 $f(0_R)=0_S$。也就是说，在环同态的作用下，一个环的加法单位元的像总是“目标”环的加法单位元。

证明 首先由群同态的定义，我们有 $f(0_R)+f(0_R)=f(0_R+0_R)$。而由环 R 上加法单位元的性质，有 $0_R+0_R=0_R$，进一步有 $f(0_R+0_R)=f(0_R)$。综合起来我们就有

$$f(0_R)+f(0_R)=f(0_R+0_R)=f(0_R)$$

给上面的等式两边加上 $-f(0_R)$：

$$(f(0_R)+f(0_R))-f(0_R)=f(0_R)-f(0_R)=0_S$$

最后一个等式使用了加法逆元的定义，利用加法结合律，我们有

$$(f(0_R)+f(0_R))-f(0_R)=f(0_R)+(f(0_R)-f(0_R))=f(0_R)+0_S=f(0_R)$$

再一次将两个等式结合起来，即得 $f(0_R)=0_S$。 ♣

- 设 $f:R\to S$ 是一个满射的环同态，假设 R 有一个乘法单位元 1_R，则 S 有一个乘法单位元 1_S，并且 $f(1_R)=1_S$。

备注 注意，不像前面有关加法单位元的讨论，这里做出了进一步的假设，即需要满射性，否则结论就可能不成立。

证明 给定 $s\in S$，，令 $r\in R$ 且使得 $f(r)=s$，则

$$f(1_R)\cdot s=f(1_R)\cdot f(r)=f(1_R\cdot r)=f(r)=s$$

因此，$f(1_R)$ 的作用就像是 S 中的单位元，由已经证明的单位元的惟一性，必有 $f(1_R)=1_S$。这就证明了命题。 ♣

备注 这里必须注意，在一个环同态 $f:R\to S$ 的作用下，乘法单位元 1_R 的像不一定就是 S 的乘法单位元 1_S。比如，我们定义一个环同态 $f:\mathbf{Q}\to S$，这里的 $\mathbf{Q}$ 为有理数环，S 为 2×2 有理数矩阵环，具体的映射为

$$f(x)=\begin{pmatrix} x & 0 \\ 0 & 0 \end{pmatrix}$$

则我们发现1的像为$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$，但是环$S$的乘法单位元却是$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$。

在一些交换环中也存在这样的例子，即在环同态的作用下单位元的像却不是单位元。比如，令$R=\mathbf{Z}/3$，$S=\mathbf{Z}/6$，定义R到S的映射如下：

$$f(r \bmod 3)=4r \bmod 6$$

我们来验证这个定义是有意义的：如果$r=r' \bmod 3$，则$3|(4r-4r')$且$2|(4r-4r')$，所以能够得到$6|4(r-r')$，即$4r=4r' \bmod 6$。我们再来验证这是一个同态：

$$f(x+y)=4(x+y)=4x+4y=f(x)+f(y)$$

实际上，这对任何数都成立，而不仅仅是4。再来看f是否保持乘法运算，实际上运用$4\cdot 4=4 \bmod 6$这一特性，我们有

$$f(x\cdot y)=4(x\cdot y)=(4\cdot 4)(x\cdot y)=(4x)\cdot(4y)=f(x)\cdot f(y)$$

因此f是一个同态。但是$f(1)=4\neq 1 \bmod 6$。

习题

29.2.01 给定整数$N>1$，详细证明由$f(x)=x+N\mathbf{Z}$给出的映射$f:\mathbf{Z}\to\mathbf{Z}/N\mathbf{Z}$是一个环同态。（我们已经知道这总是成立的。）

29.2.02 设$f:R\to S$是一个满射的环同态（为简化R,S为交换的），令I是R的一个理想，证明$J=\{f(i):i\in I\}$是S的一个理想。

29.2.03 设$f:R\to S$是一个环同态（为简化R,S为交换的），令J是I的一个理想，证明$I=\{i\in I\in R:f(i)\in J\}$是$S$的一个理想。

29.2.04(*) 证明2×2有理数矩阵环R中的两个双边理想只可能是{0}和整个环R本身。

29.3 商环

现在我们来给出一种由已知的环构造新环的方法，其中的一种特殊情况就是由整数环$\mathbf{Z}$构造$\mathbf{Z}/n$。设R是一个单位元为1的交换环，I是R的理想，**商环**R/I（R模I）定义为陪集的集合：

$$r+I=\{r+i:i\in I\}$$

我们定义R/I上的加法和乘法运算如下：

$$(r+I)+(s+I)=(r+s)+I$$

$$(r+I)\cdot(s+I)=(r\cdot s)+I$$

这个商环中的零元素为$0_{R/I}=0+I$，而单位元则为$1_{R/I}=1+I$。

例子 最常见的例子为$\mathbf{Z}/n$，只不过这里的理想I为$n\cdot\mathbf{Z}$。

但是，正如我们必须验证$\mathbf{Z}/n$上的加法和乘法运算是明确定义的一样，在这里我们也必须验证R/I上的加法和乘法运算是明确定义的。有一个问题就是这里对集合$r+I$有不同的叫法，而我们需要所谓的乘法和加法不依赖于陪集的叫法，即仅关注它们的真正含义。这也正是定义的明确性所要求的。

假设$r+I=r'+I$，$s+I=s'+I$，也就是有两个陪集，每一个均以不同的方式来表示，

为了证明所定义的加法是明确的，我们需要验证：

$$(r+s)+I=(r'+s')+I$$

而为了证明我们所给出的乘法定义的明确性则需要验证：

$$(r\cdot s)+I=(r'\cdot s')+I$$

因为 $r+I=r'+I$，特别地，$r'=r'+0\in r+I$，所以 r' 可以被表示为 $r'=r+i$，对某个 $i\in I$。类似地，对某个 $j\in I$，$s'=s+j$。于是有

$$(r'+s')+I=(r+i+s+j)+I=(r+s)+(i+j+I)$$

和 $k=i+j$ 为 I 中的元素。我们断言对任何 $k\in I$，有 $k+I=I$。我们来证实这个断言，因为 I 对加法是封闭的，所以 $k+I\subset I$。另一方面，对任意的 $x\in I$，可以将其表示为 $x=k+(x-k)$，而 $x-k\in I$，所以 $k+I\supset I$。因此 $k+I=I$。由此，

$$(r'+s')+I=(r+s)+I$$

这就证明了商环上加法的定义是明确的。类似地，我们来考虑乘法：

$$(r'\cdot s')+I=(r+i)\cdot(s+j)+I=(r\cdot s)+(rj+si+I)$$

因为 I 一个理想，所以 rj 和 si 均 $\in I$，那么 $rj+si\in I$。因此，同前面讨论加法时一样，$rj+si+I=I$。由此，

$$(r\cdot s)+I=(r'\cdot s')+I$$

所以商环上的乘法的定义也是明确的。

$0+I$ 是商环的零元素以及 $1+I$ 是它的单位元的证明是类似的。

类似地，我们也有**商同态** $q:R\to R/I$，这个映射自然就是 $q(r)=r+I$。事实上刚才的讨论已经证明了这一点。

命题 对于一个交换环 R 及其理想 I，商映射 $R\to R/I$ 是一个环同态。 ♣

29.4 极大理想和域

现在我们来考虑用极大理想的商构造域。这是一种基本的构造方法。设 R 是一个单位元为1的交换环，R 中的一个理想 M 是极大理想，如果 $M\neq R$ 并且如果对其他任意理想 $I\supset M$，则必有 $I=R$。也就是说，M 是极大理想，如果除了 R 本身以外不存在严格地大于 M（包含 M）的理想。

命题 对于一个有单位元的交换环 R，I 是 R 的一个理想，则商环 R/I 为一个域当且仅当 I 是一个极大理想。

证明 设 $x+I$ 是 R/I 的一个非零元素，则 $x+I\neq 0$，故 $x\notin I$。注意到理想 $Rx+I$ 因此严格地大于 I。因为 I 已经是极大理想，则必有 $Rx+I=R$。因此存在 $r\in R$ 和 $i\in I$，使得 $rx+i=1$，将这个等式模 I 我们得到 $rx\equiv 1 \bmod I$。这也就是说 $r+I$ 是 $x+I$ 的乘法逆元素，因此 R/I 为一个域。

另一方面，假设 R/I 为一个域。设 $x\in R$ 但 $x\notin I$，则在 R/I 中 $x+I\neq 0$。因此它在 R/I 中就有一个乘法逆元 $r+I$，即

$$(r+I)\cdot(x+I)=1+I$$

由商环中乘法的定义，上式即为 $rx+I=1+I$，或者 $1\in rx+I$，这就意味着理想 $Rx+I$ 就是 R。但是 $Rx+I$ 已经是包含 I 和 x 的最小理想，因此不存在严格大于 I 的真正理想，故 I 是极大理想。 ♣

习题

29.4.01 在系数为 $\mathbf{Z}$ 中且只有一个变元的多项式环 $\mathbf{Z}[x]$中，通过证明由 x 生成的理想不包含素数 2 进而证明它不是极大理想。

29.4.02 在系数为 $\mathbf{Z}$ 中且只有一个变元的多项式环 $\mathbf{Z}[x]$中，通过证明商环是一个域进而证明由 x 和一个素数 p 生成的理想是极大理想。

29.4.03 在系数为 $\mathbf{Z}$ 中且只有一个变元的多项式环 $\mathbf{Z}[x]$中，通过证明商环是一个域进而证明由 x^2+1 和素数 7 生成的理想是极大理想。

29.4.04 在系数为 $\mathbf{F}_2$ 中且有两个变元的多项式环 $\mathbf{F}_2[x,y]$中，通过证明由 x 生成的理想不包含 y 进而证明它不是极大理想。

29.4.05 在系数为 $\mathbf{F}_2$ 中且有两个变元的多项式环 $\mathbf{F}_2[x,y]$中，通过证明商环是一个域进而证明由 x 和 y 生成的理想是极大理想。

29.5 域扩张的更多知识

在这一节，我们将使前面讨论的构造问题更加具体，我们将用系数域“较小”的多项式环的商来获得“更大”的域。这也是我们早些时候所讨论的有限域的略微抽象的形式。设 k 是一个域，另一个包含 k 的域 K 称为 k 的**扩域**，而 k 则是 K 的一个子域。

定理 设 k 是一个域，$P(x)$ 是 $k[x]$ 中的一个不可约多项式（不是零多项式），则主理想 $I=k[x]\cdot P(x)$ 是极大理想。因此商环 $k[x]/I$ 是一个域。更进一步，复合映射

$$k \to k[x] \to k[x]/I$$

是单射，所以我们可以将 k 作为 $k[x]/I$ 的一个子域来考虑。若令 $\alpha=x+I$ 为不定元 x 在 $k[x]/I$ 中的像，则（在商 $k[x]/I$ 中）$P(\alpha)=0$。最后，任何 $\beta\in k[x]/I$ 均可惟一地表示为 $\beta=R(\alpha)$ 的形式，这里 R 是 $k[x]$ 中一个次数严格小于 $P(x)$ 的次数的多项式。

备注 k 的扩域 K 的**次数**就是构造过程中所用的多项式 $P(x)$ 的次数。

备注 在构造过程中，我们认为 α 是“存在的”，并且是方程 $P(x)=0$ 的一个根，我们称将 $P(x)=0$ 的一个根**添加**到 k 中，记为 $k[\alpha]=k[x]/I$。

备注 为了使记号简便起见，通常将商 $k[x]/k[x]\cdot P(x)$ 记为 $k[x]/P(x)$，其含义就是商是经由 $P(x)$ 生成的理想而产生的。这与我们前面的记法，如用 $\mathbf{Z}/n$ 表示 $\mathbf{Z}/\mathbf{Z}\cdot n$ 是一致的。

备注 一个可以被表示为多项式 $R(\alpha)$ 的元素 $\beta\in k[x]/I$ 是**可约简的**，这里 R 的次数严格小于 $P(x)$ 的次数。因为 $k[x]/I$ 是环，任何 α 的多项式 $R(\alpha)$ 也能表现 $k[x]/I$ 中的特征。它的每个元素均可惟一地表示为一个次数严格小于 P 的次数的多项式。这与商环 $\mathbf{Z}/n$ 上的情况完全类似，$\mathbf{Z}/n$ 中每一个等价类均在模 n 约简的整数中有一个惟一的代表元。记作{0，1，2，…，n-1}

证明 设 J 是一个多项式，但不属于理想 $I=k[x]\cdot P$，我们需要证明理想 $k[x]\cdot J+I$ 就是 $k[x]$，从而证明 I 的极大性。因为 P 是不可约的，则 J 和 P 的最大公因子就是 1。因此，由 $k[x]$ 上的欧几里得算法，存在 $k[x]$ 中的多项式 A,B，使 $A\cdot P+B\cdot J=1$。这也就是说 $k[x]\cdot J+I$ 包含 1。令 C,D 为满足 $1=C\cdot J+D\cdot P$ 的多项式，则对任何多项式 M，就有

$$M=M\cdot 1=M\cdot(C\cdot J+D\cdot P)=(M\cdot C)\cdot J+(M\cdot D)\cdot P$$

属于 $k[x]\cdot J+k[x]\cdot P$。即 M 属于理想 $k[x]\cdot J+k[x]\cdot P$，所以这个理想就是整个环 $k[x]$。这就

证明了 $I=k[x]\cdot P$ 的极大性。

下面我们来证明复合映射 $k\to k[x]\to k[x]/k[x]\cdot P$ 是一个单射。令 $I=k[x]\cdot P$，第一个映射是很显然，它将 $a\in k$ 映射为一个“常数”多项式 a。假设 $a,b\in k$，使得 $a+I=b+I$，则通过减法，$(a-b)+I=0+I$，由此即得

$$a-b=(a-b)+0\in(a-b)+I=I$$

因此 $a-b\in I$。

接下来我们证明 $P(\alpha)=0$。设 $q:k[x]\to k[x]/I$ 是一个环同态，并设

$$P(x)=a_nx^n+a_{n-1}x^{n-1}+\ldots+a_2x^2+a_1x+a_0$$

为证明 $P(\alpha)=0$，我们首先来计算

$$\begin{aligned}P(\alpha)&=a_n\alpha^n+a_{n-1}\alpha^{n-1}+\ldots+a_2\alpha^2+a_1\alpha+a_0\\&=a_nq(x)^n+a_{n-1}q(x)^{n-1}+\ldots+a_2q(x)^2+a_1\alpha+a_0\\&=q(a_nx^n+a_{n-1}x^{n-1}+\ldots+a_2x^2+a_1x+a_0)=q(P(x))\end{aligned}$$

这是因为 q 是环同态，并且 k 中的“常数”被映射到商时不会改变。因为 $P(x)\in I$，则像 $q(P(x))$ 为 0，也就是 $P(\alpha)=0$。

最后我们来证明商环 $k[x]/I$ 中的任何元素都可表示为次数不超过 $\deg P$ 的 $\alpha=x+I$ 的多项式。事实上，给定 $\beta\in k[x]/I$，存在一个多项式 J 使得 $q(J(x))=\beta$，对多项式 J 运用除法算法，可得

$$J(x)=Q(x)\cdot P(x)+R(x)$$

这里 $\deg R<\deg P$。然后对上式两边同时作用以环同态 q，我们利用 $q(P(x))=P(\alpha)=0$ 以及环同态的性质可以得到

$$\beta=q(J(x))=q(Q(x))\cdot q(P(x))+q(R(x))=q(Q(x))\cdot 0+R(q(x))=R(\alpha)$$

即 $\beta=R(\alpha)$。至此定理得证。 ♣

推论 当 k 为包含 q 个元素有限域时，对于一个 n 次不可约多项式 $P(x)$，域扩张 $K=k[x]/P(x)$ 则有 q^n 个元素。

证明 设 α 是 x 在 K 中的像，因为 K 中的每个元素均可惟一表示为次数不超过 n 的 α 的多项式 $R(\alpha)$，而对 n 个系数（α 的方幂 0 到 $n-1$ 次）中的每一个均有 q 种可能，则总共就有 q^n 个元素。 ♣

备注 由不可约多项式 $P(x)$ 所形成的域扩张 $k[x]/P(x)$，如果 $P(x)$ 为二次的，则称其为**二次的**；如果 $P(x)$ 为三次的，则称其为**三次的**；如果 $P(x)$ 为五次的，则称其为**五次的**，等等。

29.6 费罗贝尼乌斯自同构

有限域的一个非常高级的结构化特征就是**费罗贝尼乌斯映射**的出现，它在域理论的研究中起着非常重要的作用。几个关于有限域的基本定理，它们虽然没有直接提到费罗贝尼乌斯映射，但在这些定理证明中却用到了费罗贝尼乌斯映射（Frobenius maps）。在本节，我们将从更加抽象的角度考虑有限域，并且尽可能不依赖有限域的特定模型，比如依赖于不可约多项式的选择的有限域模型。实际上，这个更加抽象的研究将用来给出关于不可约多项式和本原多项式的计数定理的证明。

在本节，我们还会简要介绍一个小域上大域的**自同构群**的一些特性。自同构群是**伽罗瓦**

理论的先驱，伽罗瓦理论(Galois theory)系统地研究了这种自同构群。

令 $k=\mathbf{F}_q=GF(q)$ 是一个有 q 个元素的有限域，其中 $q=p^n$ 是素数 p 的方幂。给定整数 $N>1$，并假设有一个较大且包含 k 的有限域 $K=\mathbf{F}_{q^N}=GF(q^N)$。$k$ 上 K 的**费罗贝尼乌斯映射**为 $\Phi(\alpha)=\alpha^q$。有时对 K 和 k 的这种模糊的引用是无坏处的，因为我们也称其为 k 上 K 的费罗贝尼乌斯自同构。原因将在下面进行阐述。因为 K 对乘法是封闭的，很显然 Φ 将 K 映射到自身。

备注 Φ 就是将 K 中的元素映射为该元素的 q 次方幂，这里 q 是 K 中一个较小的域 k 的基数。我们所使用的符号记法并没有说明相应的 k 和 K 到底是什么。对 K 和 k 的这种模糊的引用是没什么坏处的，因为我们不会对 Φ 直接进行操作。

命题 $k=\mathbf{F}_q$ 上 $K=\mathbf{F}_{q^N}$ 的费罗贝尼乌斯映射 Φ 是 K 到 K 的一个双射。特别地，

$$\Phi^N=\underbrace{\Phi\circ\Phi\circ\cdots\circ\Phi}_{N}$$

是 K 上的单位映射（即将每个元素映射到自身）。

证明 因为费罗贝尼乌斯映射 Φ 就是将 K 中的元素映射为该元素的 q 次方幂，而且 K 对乘法是封闭的，因此 Φ 将 K 映射到 K。我们所关注的单射性和满射性，可以这样获得。因为如果能够证明 Φ^N 是 K 上的单位映射，则足以证明 Φ 既是单射又是满射。

由于乘法群 $\mathbf{K}^\times$ 的阶为 q^N-1，所以由拉格朗日定理及其推论，任何元素 $\beta\in\mathbf{K}^\times$ 的阶则为 q^N-1 的因子，那么 $\beta^{q^N-1}=1$。因此，对 K 中的一个非零元素 β，我们有

$$\Phi(\beta)=\beta^{q^N}=\beta\cdot\beta^{q^N-1}=\beta\cdot 1=\beta$$

这就证明了命题。 ♣

命题 限制于 k 上的费罗贝尼乌斯映射 Φ 是一个单位映射。也就是说，对于 $k=\mathbf{F}_q$ 中的每个元素 α，$\Phi(\alpha)=\alpha$。如果对于 $\alpha\in K$ 也有性质 $\Phi(\alpha)=\alpha$，则事实上 $\alpha\in k$。

证明 该命题的前面一半正好适用拉格朗日定理的推论。首先因为乘法群 $k^\times$ 的非零元素共有 $q-1$ 个，由拉格朗日定理及其推论，k 中任何非零元素的阶都是 $q-1$ 的一个因子。所以对非零元素 $\alpha\in k$，我们有

$$\Phi(\alpha)=\alpha^q=(\alpha)^{q-1}\cdot\alpha=1\cdot\alpha=\alpha$$

对于0，当然有 $0^q=0$。这就证明了命题的前半部分。

现在我们假设 $\alpha\in K$ 且 $\Phi(\alpha)=\alpha$，由映射 Φ 的定义，α 就是方程 $x^q-x=0$ 在域 K 中的一个根。由多项式的惟一分解（系数在域中），我们知道次数为 q 的多项式方程至多在域中有 q 个根。现在我们已经找到了这个方程的 q 个根，也就是包含在 K 中的一个小的域 k 的所有元素。因此在域 k 之外不可能存在这个方程的任何根，即必有 $\alpha\in k$。这就证明了命题的后半部分。 ♣

引理 设 1_K 是 K 中的乘法单位元，则 $\underbrace{1_K+\cdots+1_K}_{p}=0$。由此，对于 K 中的任何元素 α，$\underbrace{\alpha+\cdots+\alpha}_{p}=0$。

备注 一般域理论的一个更加系统化的拓展就是使这个引理的结果更加易于理解，但它所花的时间远远超过如下有趣的证明。

证明　由拉格朗日定理及其推论，在由 K 中的加法所构成的群中（暂时不考虑它的乘法），任何元素 α 的阶是整个群的阶 q^N 的一个因子，也就是有

$$q^N \cdot \alpha = \underbrace{\alpha + \cdots + \alpha}_{q^N} = 0$$

因为 $q = p^n$，上式则变为 $p^{nN} \cdot \alpha = \underbrace{\alpha + \cdots + \alpha}_{p^{nN}} = 0$。此时我们使用记号 $t = \underbrace{1_K + \cdots + 1_K}_{p}$，并把 $\alpha = 1_K$ 代入上式，则有

$$0 = \underbrace{1_K + \cdots + 1_K}_{q^N} = t^{nN}$$

因为 K 是域，不论何时几个元素乘积为 0，必有一个因子为 0。因此 $t = 0$，引理的第一部分得证。再由下式即可得引理的第二部分结论：

$$\underbrace{\alpha + \cdots + \alpha}_{p} = \underbrace{1_K \cdot \alpha + \cdots + 1_K \cdot \alpha}_{p} = \underbrace{(1_K + \cdots + 1_K) \cdot \alpha}_{p} = 0 \cdot \alpha = 0$$

至此，引理得证。 ♣

命题　对任意 $\alpha, \beta \in K$，k 上 K 的费罗贝尼乌斯映射 Φ 具有如下的性质：

$$\Phi(\alpha + \beta) = \Phi(\alpha) + \Phi(\beta)$$

$$\Phi(\alpha \cdot \beta) = \Phi(\alpha) \cdot \Phi(\beta)$$

这也就是说，Φ 保持加法和乘法。因为我们已经知道 Φ 是一个双射，所以 Φ 被称为**环同构**。

证明　第二个性质，即保持乘法，可由我们已知的简单事实得出：即一个乘积的 q 次方幂等于 q 次方幂的乘积。只要乘法是交换的，这个结论总是正确的，这里也不例外。也不用担心特定的指数到底是什么。

而保持加法的证明则需要灵活运用指数 q 是一个素数 p 的方幂这一事实。对任意 $\alpha, \beta \in K$，我们从证明 $(\alpha + \beta)^p = \alpha^p + \beta^p$ 开始，由二项式展开定理，左边即为：

$$\alpha^p + \binom{p}{1}\alpha^{p-1}b + \binom{p}{2}\alpha^{p-2}b^2 + \cdots + \binom{p}{p-1}\alpha^1 b^{p-1} + \beta^p$$

这里 K 中的元素乘以正整数就是重复的加法。如同费马小定理的证明，这个二次项展开式中所有中间项的系数均能被 p 整除，由刚刚证明的引理，在 K 中展开式中所有中间项即为 0。由此我们就证明了 $(\alpha + \beta)^p = \alpha^p + \beta^p$。然后我们再反复利用这个结果：

$$(\alpha + \beta)^{p^2} = (\alpha^p + \beta^p)^p = a^{p^2} + b^{p^2}$$

$$(\alpha + \beta)^{p^3} = (\alpha^p + \beta^p)^{p^2} = (a^{p^2} + b^{p^2})^p = \alpha^{p^3} + \beta^{p^3}$$

等等，以此类推，由归纳法我们即得 $(\alpha + \beta)^{p^{nN}} = \alpha^{p^{nN}} + \beta^{p^{nN}}$。即费罗贝尼乌斯映射保持加法。 ♣

命题　设 $P(x)$ 是系数为 $k = \mathbf{F}_q$ 上的多项式，令 $\alpha \in K$ 是方程 $P(x) = 0$ 的一个根，则 $\Phi(\alpha) = \alpha^q$，$\Phi^2(\alpha) = \Phi(\Phi(\alpha)) = \alpha^{q^2}$，…均是方程的根。

证明　设系数 C_i 为 $k = \mathbf{F}_q$ 上的多项式 $P(x)$ 为：

$$P(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_2 x^2 + c_1 x + c_0$$

对如下方程两边同时取费罗贝尼乌斯映射

$$0 = c_n \alpha^n + c_{n-1}\alpha^{n-1} + \cdots + c_2 \alpha^2 + c_1 \alpha + c_0$$

则有

$$\Phi(0)=\Phi(c_n)\Phi(\alpha)^n+\Phi(c_{n-1})\Phi(\alpha)^{n-1}+\cdots+\Phi(c_2)\Phi(\alpha)^2+\Phi(c_1)\Phi(\alpha)+\Phi(c_0)$$

因为 Φ 在 K 中保持加法和乘法，而且系数 c_i 在 k 中，如同左边的 0 一样，Φ 不改变它们。即我们得到

$$0=c_n\Phi(\alpha)^n+c_{n-1}\Phi(\alpha)^{n-1}+\cdots+c_2\Phi(\alpha)^2+c_1\Phi(\alpha)+c_0$$

这也就是 $0=P(\Phi(\alpha))$。因此如果 α 是方程 $P(x)=0$ 的一个根，则 $\Phi(\alpha)$ 也是它的一个根。反复利用这一结论即得我们的命题。 ♣

命题 令 $A=\{\alpha_1,\cdots,\alpha_t\}$ 是 K 中（t 个不同）元素的集合，并且满足对任意的 $\alpha\in A$，$\Phi(\alpha)\in A$，则多项式 $(x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_t)$ 的系数在 k 中。

证明 对系数在大域 K 中的一个多项式

$$P(x)=c_nx^n+c_{n-1}x^{n-1}+\cdots+c_2x^2+c_1x+c_0$$

让 Φ 作用于该多项式的系数，定义一个新的多项式 $\Phi(P)$：

$$\Phi(P)(x)=\Phi(c_n)x^n+\Phi(c_{n-1})x^{n-1}+\cdots+\Phi(c_2)x^2+\Phi(c_1)x+\Phi(c_0)$$

因为 Φ 在 K 中保持加法和乘法，故不难验证它对系数在 K 中的多项式也保持加法和乘法，即对两个这样的多项式 P 和 Q，有如下性质成立：

$$\Phi(P+Q)=\Phi(P)+\Phi(Q)$$

$$\Phi(P\cdot Q)=\Phi(P)\cdot\Phi(Q)$$

给多项式乘积 $(x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_t)$ 作用以 Φ，由关于 Φ 的假设，即它只是对 A 中的元素做一置换，这样它也就只是打乱了多项式 $(x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_t)$ 中乘积因子的顺序，故

$$\Phi((x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_t))=(x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_t)$$

这就意味着 $(x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_t)$ 的展开形式

$$(x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_t)=c_nx^n+c_{n-1}x^{n-1}+\cdots+c_2x^2+c_1x+c_0$$

具有如下性质：

$$\begin{aligned}&c_nx^n+c_{n-1}x^{n-1}+\cdots+c_2x^2+c_1x+c_0\\&=\Phi(c_n)x^n+\Phi(c_{n-1})x^{n-1}+\cdots+\Phi(c_2)x^2+\Phi(c_1)x+\Phi(c_0)\end{aligned}$$

多项式“相等”的含义就是对应项的系数相等，因此由上面的多项式等式我们可得 $\Phi(c_i)=c_i$，对所有下标 i。这就意味着对所有的下标 i，$c_i\in k$。 ♣

命题 设 α 是 $K=k[x]/Q$ 的一个元素，则存在 $k[x]$ 中的一个首一不可约多项式 P 使得 α 是 $P(x)=0$ 的一个根，即

$$P(x)=(x-\alpha)(x-\Phi(\alpha))(x-\Phi^2(\alpha))\cdots(x-\Phi^{d-1}(\alpha))$$

这里的 d 是满足 $\Phi(\alpha)=\alpha$ 的最小正整数。

证明 考虑 α 在费罗贝尼乌斯映射下连续的像 $\Phi^i(\alpha)$。因为域是有限的，所以存在某个 $0\leqslant i<j$ 使得点 $\Phi^i(\alpha)=\Phi^j(\alpha)$。因为 Φ 是 K 到 K 的双射，它必有一个逆映射 Φ^{-1}。对等式 $\Phi^i(\alpha)=\Phi^j(\alpha)$ 作用逆映射 Φ^{-1} i 次，则得到

$$\alpha=\Phi^0(\alpha)=\Phi^{j-i}(\alpha)$$

即有 $i=0$。这意味着在 $\alpha,\Phi(\alpha),\Phi^2(\alpha),\cdots$ 这个列表中，对最小的 j，使得 $\Phi^j(\alpha)$ 已经在列表上了，我们有 $\Phi^j(\alpha)=\alpha$，而不用在列表上进一步复制某个其他元素了。令

$$\alpha,\Phi,\cdots,\Phi^{d-1}(\alpha)$$

是α在费罗贝尼乌斯映射下不同的像。我们看到有$\Phi^d(\alpha)=\alpha$。令

$$P(x)=(x-\alpha)(x-\Phi(\alpha))(x-\Phi^2(\alpha))\cdots(x-\Phi^{d-1}(\alpha))$$

同前面一样，对P运用Φ，仅仅是通过指标的移位对右边的因子作一个置换。因此当展开多项式P后，它没有变化，所以它的系数仍然在小域k中。在前面讨论费罗贝尼乌斯映射时已经看到了这种现象。显然α是$P(x)=0$的一个根。

由上面的讨论，如果β是一个系数在小域k上的多项式方程在K中的一个根，则$\Phi(\beta)$也是一个根。因此对任何系数在小域k上的多项式，如果α是它的一个根，则它必有因子$x-\Phi^i(\alpha)$，$1\leqslant i<d$。由系数在域上的多项式的惟一分解，这表明这是惟一的此类多项式。

特别地，P必定是$k[x]$中的一个不可约多项式，因为如果它能够在$k[x]$被真正分解为$P=P_1P_2$，则由惟一分解，α或者是$P_1(x)=0$的根，或者是$P_2(x)=0$的根，并且所有的d个不同的元素$\Phi^i(\alpha)$也是同一方程的根。因为根的个数至多为多项式的次数，因此不可能有P的真分解，所以P在$k[x]$是不可约的。 ♣

推论 设β是x在$K=\mathbf{F}_q[x]/Q$中的像，则

$$Q(x)=(x-\beta)(x-\Phi(\beta))(x-\Phi^2(\beta))\cdots(x-\Phi^{n-1}(\beta))$$

我们有$\Phi^n(\beta)=\beta$，并且n是使得$\Phi^n(\beta)=\beta$成立的最小正整数。

证明 这个结论就是刚才已证明命题的特殊情形。 ♣

我们还需要作进一步的抽象。令e表示$K=\mathbf{F}_q[x]/Q$到自身的单位映射，并且令

$$G=\{e,\Phi,\Phi^2,\cdots,\Phi^{n-1}\}$$

这里的Q是n次的。这是一个$K=\mathbf{F}_q[x]/Q$到自身映射的集合，如果将这些映射中的每一个均限制地$\mathbf{F}_q$上，则得到$\mathbf{F}_q$到自身的单位映射。因为每个Φ^i是$\mathbf{F}_q$上的单位映射并且将K一一地映射到自身，我们称G是$\mathbf{F}_q$上K的**自同构**。

命题 F_q上K的自同构集合G是一个以e为单位元的群。

证明 设β是x在K中的像。我们首先来验证对任意的$\alpha\in K$，$\Phi^n(\alpha)=\alpha$。我们已经知道α可以表示为一个系数在F_q上β的多项式，即$\alpha=R(\beta)$。因为Φ保持加法和乘法，并且由前面的推论，有$\Phi^n(\beta)=\beta$，因此

$$\Phi^n(\alpha)=\Phi^n(R(\beta))=R(\Phi^n(\beta))=R(\beta)=\alpha$$

因为$\Phi^n=e$，对任意的整数 ，$\Phi^l=\Phi^{l\%n}$是K到自身的一个函数，这里的$\%n$就是通常的模n约简。因此集合G对乘法是封闭的（也就是函数的合成）

$$\Phi^i\circ\Phi^j=\Phi^{(i+j)\%n}$$

进一步我们可以证明G对求逆是封闭的：$(\Phi^i)^{-1}=\Phi^{n-i}$。结合律可以由函数合成的结合律直接得到。群的单位元为映射e是显然的。 ♣

对于$\alpha\in K$，α在G中的**稳定化子子群**G_α定义为$G_\alpha=\{g\in G:g(\alpha)=\alpha\}$，我们容易验证这是$G$的一个子群。

命题 对于$\alpha\in K$，α在G中的稳定化子子群G_α是G的一个子群。

证明 G_α显然包含单位映射e。假设$g\in G$且满足$g(\alpha)=\alpha$，对这个等式运用函数g^{-1}则得到

$$\alpha=g^{-1}(g(\alpha))=g^{-1}(\alpha)$$

因为G_α对求逆是封闭的。如果$g,h\in G_\alpha$，则$g(h(\alpha))=g(\alpha)=\alpha$。因此$G_\alpha$对乘法是封闭的，

所以 G_α 的确是一个群。 ♣

命题 给定 $\alpha \in K = \mathbf{F}_q[x]/Q$，则 α 在反复运用费罗贝尼乌斯映射下不同像 $\Phi^i(\alpha)$ 的个数是 Q 次数的因子。

证明 我们在这里真正应当断言的是：不同像 $\Phi^i(\alpha)$ 的集合与陪集 G/G_α 的集合是双射，G_α 是 α 在自同构群 G 中的稳定化子子群。实际上，如果 $g \in G$，并且 $h \in G_\alpha$，则 $(gh)(\alpha) = g(h(\alpha)) = g(\alpha)$。这就证明 $gG_\alpha \to g(\alpha)$ 是定义明确的。并且如果 $g(\alpha) = g'(\alpha)$，则 $\alpha = g^{-1}g'(\alpha)$，因此 $g^{-1}g' \in G_\alpha$。因此我们就证明了不存在两个不同的陪集 gG_α 和 $g'G_\alpha$ 把 α 映射为相同的元素。 ♣

推论 设 α 属于域 $K = k[x]/Q$，则使得 $P(\alpha) = 0$ 的系数在 k 中惟一的首一不可约多项式 P 的次数是 Q 的次数 n 的因子。

证明 由前面的讨论，设 $P(x) = (x-\alpha)(x-\Phi(\alpha))(x-\Phi^2(\alpha))\cdots(x-\Phi^{d-1}(\alpha))$，这里的 $\alpha, \Phi(\alpha), \Phi^2(\alpha), \cdots, \Phi^{d-1}(\alpha)$ 为 α 的不同的像，d 是多项式 P 的次数。如同我们在拉格朗日定理及其推论的证明过程一样，所有的陪集 G_α 有相同的基数。由此和前面的命题，$\mathrm{card}(G) = d \cdot \mathrm{card}(G_\alpha)$。同样，在 x 为 β 在 K 中的像的特殊情形，稳定化子子群正好就是 $\{e\}$，所以 $\mathrm{card}(G) = n \cdot 1$，因此 d 是 n 的因子。 ♣

备注 在本节的讨论中，我们没有对有限域 $\boldsymbol{F}_q$ 和 $\boldsymbol{F}_{q^N}$ 的惟一性做出任何假设，我们只是用这样的记法来提醒有限域 k 和 K 的元素个数。

习题

29.6.01 S 为一个集合，$f: S \to S$ 是 S 到自身的一个函数。假设 $f \circ f$ 是 S 上的一个单位映射，详细证明 f 是一个双射。

29.6.02 S 为一个集合，$f: S \to S$ 是 S 到自身的一个函数。假设 $f^N = \underbrace{f \circ \cdots \circ f}_{N}$ 是 S 上的一个单位映射，详细证明 f 是一个双射。

29.6.03 S 为一个有限集合，证明如果 $f: S \to S$ 是一个单射，则它必定是一个满射；如果 f 是一个满射，则它必定是一个单射。

29.7 不可约多项式的计数

知道有限域上存在本原根之后，我们就可以对给定次数的不可约多项式进行计数。我们将充分利用已经掌握的费罗贝尼乌斯映射的基本结论。在这个过程中，我们也会不可避免地了解并掌握更多的关于有限域的结构特性。

一个整数称为是**无平方的**，如果它不能被任何素数的平方整除。我们来回忆一下对整数所定义的墨比乌斯函数 μ：

$$\mu(n) = \begin{cases} (-1)^t & \text{如果 } n \text{ 是无平方的，即恰被 } t \text{ 个素数整除} \\ 1 & \text{如果 } n = 1 \\ 0 & \text{如果 } n \text{ 被某个素数的平方整除} \end{cases}$$

定理 $\mathbf{F}_q[x]$ 上次数为 n 的首一不可约多项式的个数为 $\dfrac{1}{n}\sum_{1 \le d \le n, d \mid n} \mu(d) q^{n/d}$。也就是说这个

数字为 $\frac{1}{n}\cdot\left(q^n-\sum_{p_1|n}q^{n/p_1}+\sum_{p_1,p_2|n}q^{n/p_1p_2}-\sum_{p_1,p_2,p_3|n}q^{n/p_1p_2p_3}+\ldots\right)$。这里求和时要取遍 n 所有不同的素因子。

我们可以对上述公式进行一些特别处理，以获得一些有趣并且便于记忆的特例：

推论 如果 $n=p_1$ 是一个素数，则 $\mathbf{F}_q[x]$ 上次数为 n 的首一不可约多项式的个数为 $\frac{q^n-q}{n}$。 ♣

推论 如果 $n=p_1p_2$ 为两个不同素数 p_1 和 p_2 的乘积，则 $\mathbf{F}_q[x]$ 上次数为 $n=p_1p_2$ 的首一不可约多项式的个数为 $\frac{q^{p_1p_2}-q^{p_1}-q^{p_2}}{p_1p_2}$。 ♣

推论 如果 $n=p_1^e$ 为素数 p_1 的方幂，则 $\mathbf{F}_q[x]$ 上次数为 $n=p_1^e$ 的首一不可约多项式的个数为 $\frac{q^{p_1^e}-q^{p_1^{e-1}}}{n}$。 ♣

设 Q 是 $\mathbf{F}_q[x]$ 上次数为 n 的首一不可约多项式，则我们知道 $K=\mathbf{F}_q[x]/Q$ 是一个有 q^n 个元素的有限域。令 $\Phi(\gamma)=\gamma^q$ 是 $k=\mathbf{F}_q$ 上 K 的费罗贝尼乌斯自同构。由前面费罗贝尼乌斯自同构的讨论，我们知道对任意的 $\alpha\in K$，存在惟一的系数在 $k=\mathbf{F}_q$ 上的首一不可约多项式 P 使得 $P(\alpha)=0$，而且事实上，

$$P(x)=(x-\alpha)(x-\Phi(\alpha))(x-\Phi^2(\alpha))\cdots(x-\Phi^{d-1}(\alpha))$$

这里的 d 是反复运用 $k=\mathbf{F}_q$ 上 K 的费罗贝尼乌斯自同构 Φ 后 α 的不同像的个数。我们有如下命题：

命题 设 P 是次数为 d 的不可约首一多项式，d 整除不可约多项式 Q 的次数 n，则 $P(x)$ 在 $K=k[x]/Q$ 中有 d 个不同的根，因此 $P(x)$ 可以 K 中分解为不同线性因子的乘积。

证明 商环 $L=k[x]/P$ 是一个域，设 α 是 x 在 $L=k[x]/P$ 中的像。我们知道 $P(\alpha)=0$，并且由前面关于费罗贝尼乌斯映射的讨论，我们知道

$$P(x)=(x-\alpha)(x-\Phi(\alpha))(x-\Phi^2(\alpha))\cdots(x-\Phi^{d-1}(\alpha))$$

由拉格朗日定理及其推论，因为 $L^\times$ 的阶为 q^d-1，所以有 $\alpha^{q^d-1}=1$。由系数在域上的多项式的惟一分解，这即意味着 $P(x)$ 整除系数在 $k=\mathbf{F}_q$ 的多项式 $x^{q^d-1}-1$。

另一方面，K 中存在本原根 g 即意味着 $g^{q^n-1}=1$ 且不存在更小的正指数使该等式成立。因此 $g^1,g^2,g^3,\ldots,g^{q^n-1}$ 是所有不同的（且非零）元素。对任何整数 t

$$(g^t)^{q^n-1}=(g^{q^n-1})^t=1^t=1$$

所以这 q^n-1 个元素是 $x^{q^n-1}-1=0$ 的所有根。而另一方面，这个方程是 q^n-1 次的，所以它至多有 q^n-1 个根。因此我们断言在 $K[x]$ 中，

$$x^{q^n-1}-1=(x-g^1)(x-g^2)(x-g^3)\cdots(x-g^{q^n-1})$$

即允许它的系数在更大的域 K 上。

因为 d 整除 n，我们有一个基本的代数等式：

$$q^n-1=(q^d-1)(q^{(n-d)}+q^{(n-2d)}+q^{(n-3d)}+\ldots+q^d+1)$$

因此 q^d-1 整除 q^n-1，再由基本代数等式 $x^{q^d-1}-1$ 整数除 $x^{q^n-1}-1$。又因为 $P(x)$ 整除 $x^{q^d-1}-1$，

进而 $P(x)$ 整除 $x^{q^n-1}-1$。因此由 $x^{q^n-1}-1$ 在 $K[x]$ 可分解为线性因子，可知 $P(x)=0$ 在 K 中有 d 个根。 ♣

证明 要证明定理的结论，我们可以通过对 K 中元素计数的方式实现，即按照系数在 $k=\mathbf{F}_q$ 上的首一不可约多项式元素的根进行分组，总共分为 d 组，这里的 d 要取遍 n 的正因子，包括1和 n。设 N_d 是次数为 d 且系数在 $k=\mathbf{F}_q$ 上的首一不可约多项式的个数，则由这样的分组和计数我们有：

$$q^n=\sum_{d|n} d\cdot N_d$$

则由墨比乌斯反演我们即得公式

$$d\cdot N_n=\sum_{d|n}\mu(d)q^{n/d}$$

这就得到了前面定理的结论。 ♣

29.8 本原多项式的计数

通过前面我们对有限域的更加详细的讨论，现在我们来证明本原多项式的计数公式，作为分圆多项式不可约因子的本原多项式的特征的一个推论。

给定一个素数方幂 q，令 $k=\mathbf{F}_q$ 是一个有 q 个元素的有限域。回忆一下本原多项式的定义：次数为 n 且系数在 k 上的一个多项式 Q 为本原多项式，如果 $x^{q^n-1}=1 \bmod Q(x)$ 并且不存在更小的正整数使这个等式成立。为了简便起见，我们记 $N=q^n-1$。

定理 $\mathbf{F}_q[x]$ 中一个 n 次本原多项式 Q 是 q^n-1 阶分圆多项式的一个不可约因子。反之，q^n-1 阶分圆多项式的每个不可约因子的次数为 n 且是本原多项式。

设 φ 是欧拉的 Φ 函数，回想一下 N 阶分圆多项式 φ_N 的次数为 $\varphi(N)$。由此我们首先给出上述定理的一个推论：

推论 在多项式环 $\mathbf{F}_q[x]$ 中有 $\varphi(q^n-1)/n$ 个次数 n 为的本原多项式。

证明 由定理可知，每个这样的本原多项式是 N 阶分圆多项式 φ_N 的不可约因子，这里 $N=q^n-1$。另一方面，φ_N 的每个不可约因子是次数为 n 的本原多项式。因为 q^n-1 与我们这里所使用的域的特征 p 是互素的，因此 φ_N 没有重因子，因此也就没有哪一个因子会重复出现。又因为乘积的次数为因子次数的和，故我们有：

$$\varphi_N\text{的次数}=(n\text{次本原多项式的个数})\cdot n$$

由此我们即得推论中的计数公式。 ♣

证明 不失一般性，我们仅考虑 $n>1$ 的情况，因为线性的情况容易处理。特别地，这也就排除了能被 x 整除的不可约多项式的情况。

一方面，假设整除 q^n-1 阶分圆多项式 φ_N 的多项式 Q 次数为 n，因为 $\varphi_N \mid x^{q^n-1}-1$，当然有 $x^{q^n-1}=1 \bmod Q(x)$。如果对任何较小的正整数 t，也有类似的性质，即

$$x^t=1 \bmod Q(x)$$

则多项式 Q 整除 x^t-1。但由对分圆多项式的讨论，因为 q^n-1 与 p 是互素的，则对于 $t<q^n-1$，φ_N 与 x^t-1 没有公因子。因此，满足使 x 的方幂模 $Q(x$ 为1的最小正整数只能是 q^n-1，即 $Q(x)$ 为 n 次本原多项式。

另一方面，假设$Q(x)$为n次的本原多项式，则由定义有

$$x^{q^n-1}=1 \bmod Q(x)$$

这等价于说$Q(x)$为$x^{q^n-1}-1$的一个因子。同样，不存在使x的方幂模$Q(x)$为1的较小正整数，这个条件也就是说对任何较小的N，即$N<q^n-1$，$Q(x)$不能整除x^N-1。由分圆多项式的讨论，并利用惟一分解，如果对于$N<q^n-1$，我们从$x^{q^n-1}-1$中去掉它与x^N-1的公因子，则余下就是分圆多项式φ_N。因此n次的本原多项式$Q(x)$整除q^n-1阶分圆多项式φ_N。

现在我们来证明φ_N的所有不可约因子的次数为n。令$Q(x)$为φ_N的任何不可约因子，则$L=\mathbf{F}_q[x]/Q$为一个域，且由分圆多项式φ_N的构造可知，x的像α在乘法群L^x中的阶为$N=q^n-1$。因此，如果d是多项式$Q(x)$的次数，则由拉格朗日定理及其推论，必有q^n-1整除q^d-1。特别地，$d\geqslant n$。即一方面，φ_N的任何不可约因子的次数至少为n。而另一方面，令Φ是L在$\mathbf{F}_q$上的费罗贝尼乌斯映射：

$$\Phi(\beta)=\beta^q$$

如果

$$x^{q^n-1}=1 \bmod Q(x)$$

则有

$$\alpha^{q^n-1}=1$$

这等价于

$$\alpha^{q^n}=\alpha$$

即$\Phi^n(\alpha)=\alpha$。由费罗贝尼乌斯自同构的讨论，这意味着$\mathbf{F}_q[x]$中满足$f(\alpha)=0$的首一不可约多项式$f(x)$至多是n次的。同时，由有限域的构造，我们还有$Q(\alpha)=0$。由费罗贝尼乌斯自同构的讨论的推论，恰好存在一个满足$f(\alpha)=0$的首一不可约多项式$f(x)$，则$Q(x)=f(x)$。因为$f(x)$至多是n次的，则$Q(x)$也是至多是n次的。因此，φ_N的所有不可约因子均是n次的，这里$N=q^n-1$。

最后，我们知道本原多项式必须是不可约多项式。实际上，$\mathbf{F}_q[x]$中的一个n次本原多项式$Q(x)$整除分圆多项式φ_N，这里$N=q^n-1$。综上所述，我们就证明了φ_N的所有不可约因子均是n次的。因此根据惟一因式分解原理，Q是不可替代的，但是可约简的。 ♣

附录A 相关公式

A.1 集合与函数

这里我们来复习一下一些关于集合和函数的基本却非常重要的术语和概念。我们使用**映射**这个词作为“函数”的同义词，一般也是这样做的。

直观地讲，**集合**就是一个用“列举出它们”或者“用规则指定它们”这样的方式来描述的一些的“事物”的整体。注意这种描述是十分不准确的，但却容易为我们所理解。我们也可以说一个**集合**就是不同事物的无序的列表。

对于我们常用的一些集合，都有标准的记号：

$\phi = \{\}$ = 空集=没有任何元素的集合

$\mathbf{Z}$ = 整数集合

$\mathbf{Q}$ = 有理数集合

$\mathbf{R}$ = 实数集合

$\mathbf{C}$ = 复数集合

一个集合可以由列举方式给出，比如 $S = \{1,2,3,4,5,6,7,8\}$，其实这就是大于 0 小于 9 的整数的集合。这个集合如果以规则指定的方式给出则是

$$S = \{1,2,3,4,5,6,7,8\} = \{x : x\text{是整数且}1 \leqslant x \leqslant 8\}$$

通常集合一般的格式和记法是这样的：$\{x : x\text{具有某种性质}\}$。

如果 x 在集合 S 中，则记为 $x \in S$ 或 $S \ni x$，并称 x 是 S 的一个**元素**。因此，一个集合就是其所有元素的整体（尽管这仅仅是字面上的解释）。值得注意的是集合中元素列举的顺序对集合本身没有影响，并且如果一个集合的元素在列举时出现重复，这不产生任何影响。比如，

$$\{1,2,3\} = \{1,1,2,3\} = \{3,2,1\} = \{1,3,2,1\}$$

一个集合 S 的**子集** T 是一个其所有元素全为 S 的元素的集合。这通常记为 $T \subset S$ 或者 $S \supset T$。因此总有 $S \subset S$ 和 $\phi \subset S$。如果 $T \subset S$ 并且 $T \neq S$ 和 $T \neq \phi$，则 T 是 S 的一个**真**子集。注意，空集是任何集合的子集。对集合 S 的一个子集 T，T（在 S 中）的**补集**为

$$T^c = S - T = \{s \in S : s \notin T\}$$

集合也可以作为其他集合的元素。比如，$\{\mathbf{Q},\mathbf{Z},\mathbf{R},\mathbf{C}\}$ 是一个有四个元素的集合，每一个元素则是我们所熟悉的数集。或者我们可以验证 $\{\{1,2\},\{1,3\},\{2,3\}\}$ 是一个由 $\{1,2,3\}$ 的两个元素的子集构成的集合。

两个集合 A,B 的**交**就是同时属于这两个集合的元素构成的集合，记为 $A \cap B$。两个集合**不相交**，如果它们的交为空集。如果交不为空，则我们可以称两个集合**相遇**。两个集合 A,B 的并就是由这两个集合中所有元素构成的集合，记为 $A \cup B$。

注意，$1 \neq \{1\}$，并且 $\{\{1\}\} \neq \{1\}$。这也就是说集合的单个元素 a 与集合 $\{a\}$ 本身是两个不

同的事物。

一个**有序对** (x,y) 就是两个元素的列表，其中一个是第一个元素，如这里的 x，有一个是第二个元素，如这里的 y。两个有序对 (x,y) 和 (x',y') **相等**，当且仅当 $x=x'$ 和 $y=y'$。

两个集合 A,B 的**笛卡尔积**就是**有序对** (a,b) 的集合，其中 $a\in A$，$b\in B$，记为 $A\times B$。因此，集合 $\{a,b\}$ 可以认为是一个无序对。因为由定义 $\{a,b\}=\{b,a\}$，顺序不影响集合元素的列举。但对有序对而言，$(a,b)\neq(b,a)$，除非 $a=b$。

在 $A=B$ 的情况，通常将笛卡尔积 $A\times B$ 记为 $\mathbf{A}^2$。更一般地，对于一个给定的整数 n，一个集合的 n 次**笛卡尔积** A^n 就是有序的 n 元组 $(a_1,a_2,\ldots,a_n)$ 的集合，这里 $a_i\in A$。

一些非常重要的笛卡尔积的例子有由 **R**、**Q** 或 **C** 构成的积，比如 $\mathbf{R}^2$ 就是有序的实数对构成的集合，它用来描述平面上的点。$\mathbf{R}^3$ 为有序的三元实数构成的集合，它用来描述三维空间上的点。

集合 S 的**幂集**就是由集合的子集所组成的集合，有时将幂集记为 $\mathcal{P}S$，因此我们有

$$\mathcal{P}\phi=\{\phi\}$$

$$\mathcal{P}\{1,2\}=\{\phi,\{1\},\{2\},\{1,2\}\}$$

直观地讲，一个由集合 A 到另一个集合 B 的**函数** f 就是一个“规则”，它将每个元素 $a\in A$ 赋值给另一个元素 $b=f(a)\in B$。通常我们将函数 f 记为 $f:A\to B$，尽管这个记号没有给出关于函数 f 的任何详细信息。

更加严格地，但也是比较直观的，我们可以通过给出图的方式定义一个函数：正式的定义就是一个函数 $f:A\to B$ 就是具有如下性质的 $A\times B$ 的一个子集：对每个 $a\in A$，存在一个惟一的 $b\in B$，使得 $(a,b)\in f$。我们可以记为 $f(a)=b$。

这个正式的定义是值得注意的，至少它说明不必绝对地要求用任何可认识或简单的“公式”来描述一个函数。

下面给出一个函数正式定义的例子，令 $f:\{1,2\}\to\{2,4\}$ 是一个乘以 2 的函数，即 $f(1)=2$，$f(2)=4$。那么“正式的”函数定义将表明 f 真正是笛卡尔积的集合 $\{1,2\}\times\{2,4\}$ 的子集，且包含两个有序对 $(1,2)$ 和 $(2,4)$。即函数 f 的正式定义就是集合 $f=\{(1,2),(2,4)\}$。当然，尽可能以比图像方式更加直观的方式给出函数的描述。

一个函数 $f:A\to B$ 是**满射**（**映上的**），如果对每个 $b\in B$，存在一个 $a\in A$ 使得 $f(a)=b$。一个函数 $f:A\to B$ 是**单射**（**一对一的**），如果 $f(a)=f(a')$ 蕴含着 $a=a'$。即 f 是单射，如果对每个 $b\in B$，至多存在一个 $a\in A$ 使得 $f(a)=b$。一个映射是**双射**，如果它既是单射又是满射。

一个集合中元素的个数称为它的**基数**。这里有一点非常重要，即两个集合 A 和 B 有相同个数的元素，即有相同的基数，当且仅当它们之间存在一个双射。也就是说，如果它们有相同个数的元素，则可以某种方式使这些元素完全配对。反之，如果元素可以配对，则两个集合的元素个数必定相同。

备注 集合基数的“计数”定义完全适用于有限集，但对无限集合却无能为力。通常，如果我们也试图对无限集进行“计数”，首先要做的就是要定义基数（按照“个数”），而不是定义有相同基数的性质：两个集合有**相同的基数**，如果它们之间存在一个双射。不定义“基数”本身而定义有相同的基数，这个策略似乎在变戏法，这使得我们不必真正去数两个集合看它们是否有相同个数的元素。这类事情本身非常有趣，但在本书中对我们不是很重要。

因为在有限集中，我们可以按照传统的方式去数元素个数，很显然一个有限集与其真子集之间不存在双射。真子集的元素个数肯定小于原集合的元素个数。

与之形成鲜明对照的是，我们很可能在一个无限集与其真子集之间找到一个双射。比如，所有自然数的集合 A 与偶自然数的集合 E 之间就有一个双射：$n \to 2n$。但 E 的确是 A 的一个真子集。甚至还可以找到比这更令人吃惊的例子。最后，我们给出无限集的一个定义，一个集合是**无限的**，如果在它本身与其真子集之间存在一个双射。

令 $f: A \to B$ 是一个由集合 A 到另一个集合 B 的函数，$g: B \to C$ 是一个由集合 B 到另一个集合 C 的函数，则**复合函数** $g \circ f$ 定义为：对 $a \in A$

$$(g \circ f)(a) = g(f(a))$$

称一个非空集合 S 上的函数 $f: S \to S$ 为**恒等函数**，如果对所有 $a \in A$，有 $f(a) = a$。通常非空集合 S 上的恒等函数记为 id_S。

设 $f: A \to B$ 是一个由集合 A 到一个集合 B 的函数，f 的一个**反函数** $g: B \to A$（如果这个 g 存在）满足如下条件：对所有的 $b \in B$ 有 $(f \circ g)(b) = b$，并且对所有的 $a \in A$ 有 $(g \circ f)(a) = a$。即反函数（如果存在）具备如下性质：

$$f \circ g = \mathrm{id}_B, \quad g \circ f = \mathrm{id}_A$$

f 的反函数如果存在，则通常记为 f^{-1}。（跟 $1/f$ 不一样！）

命题 一个由集合 A 到一个集合 B 的函数 $f: A \to B$ 有反函数，当且仅当 f 是双射。此时，反函数是惟一的（即仅有一个反函数）。

证明 我们来定义一个函数 $g: B \to A$：给定 $b \in B$，令 $a \in A$ 满足 $f(a) = b$，则定义 $g(b) = a$。对每个 $b \in B$，重复上述步骤。注意到我们可利用满射性可知道对每个 b，存在一个 a，而利用单射性则可保证它的惟一性。

为验证 $g \circ f = \mathrm{id}_A$，我们做如下计算：首先，对任意 $a \in A$，$f(a) \in B$。那么由定义，$g(f(a))$ 就是满足 $f(a) = f(a')$ 的一个元素 $a' \in A$。因为 f 是单射，则必有 $a' = a$。为验证 $f \circ g = \mathbf{1}$，取 $b \in B$ 并计算：由 g 的定义，$g(b)$ 是满足 $f(g(b)) = b$ 的一个属于 A 的元素。这样我们就证明了命题。 ♣

习题

A.1.01 下列集合中各有多少个元素？$A = \{1,2,2,3,3,4,5\}$，$B = \{1,2,\{2\},3,\{3\},4,5\}$，$C = \{1,2,\{2,3\},3,4,5\}$？

A.1.02 设 $A = \{1,2,3,4,5\}$，$B = \{3,4,5,6,7\}$，列出集合 $A \cup B$、$A \cap B$ 以及 $\{x \in A : x \notin B\}$ 中的元素。

A.1.03 列出集合 $\{1,2,3\}$ 的幂集中的所有元素。

A.1.04 设 $A = \{1,2,3\}$，$B = \{2,3\}$，列出笛卡尔积 $A \times B$ 中的所有元素（不重复）。

A.1.05 由集合 $\{1,2,3\}$ 到集合 $\{2,3,4,5\}$ 之间有多少个函数？

A.1.06 由集合 $\{1,2,3\}$ 到集合 $\{1,2,3,4\}$ 之间有多少个单射函数？

A.1.07 由集合 $\{1,2,3,4\}$ 到集合 $\{1,2,3\}$ 之间有多少个满射函数？

A.1.08 如果函数 $f: A \to B$ 和函数 $g: B \to C$ 均有反函数，证明 $g \circ f$ 也有一个反函数，而且该反函数就是 $f^{-1} \circ g^{-1}$。

A.1.09 证明对一个满射函数 $f: A \to B$ 存在一个**右反函数** g，即存在一个函数 $g: B \to A$

使得 $f \circ g = \mathrm{id}_B$（但不要求 $g \circ f = \mathrm{id}_A$）。

A.1.10 证明对一个单射函数 $f: A \to B$ 存在一个**左反函数** g，即存在一个函数 $g: B \to A$ 使得 $g \circ f = \mathrm{id}_A$（但不要求 $f \circ g = \mathrm{id}_B$）。

A.1.11 给出一个偶整数集合 $2\mathbf{Z}$ 到所有整数集合 $\mathbf{Z}$ 的一个双射。

A.1.12(*) 给出一个由所有整数集合到非负整数集合的一个双射。

A.1.13(**) 给出一个由所有正整数集合到所有有理数集合的一个双射。

A.1.14(**) 这里举例说明在形成集合时，使用过于幼稚的“规则”会带来风险。设 S 为不是自己的一个元素的所有集合的集合，即：S={集合 x：$x \notin x$}证明是否有 $S \in S$ 或 $S \notin S$?(提示：假设 S 是或者不是它本身的一个元素，推导出一个矛盾的结论。如何证明？)

A.2 搜索与排序

基本的排序和搜索算法中的效率是许多其他算法的基本组成部分。我们将介绍一些通常速度较慢的排序算法（如*插入排序*、*选择排序*和*冒泡排序*），然后再介绍几个对普通数据获得较好性能的快速排序算法（*合并排序*和*快速排序*）。这个简短的讨论当然不能公正地评价所有可能的排序，但我们可以从中领略一些重要的思想。

特别地，为了确保各种可计算算法以其所声称的速度运行，有必要了解如下两点：

- 一个长为 n 数表排序好要使用 $O(n)$ 次比较；
- 在一个长为 n 且已经排序好的数表上找到指定有元素（或者证实不在该数表上）需要使用 $O(\log n)$ 次比较。

线性搜索。假设有一个整数表 $L = \{t_1, t_2, \cdots, t_n\}$，我们的问题是：*给定另一个整数 t，寻找 $t_i \in L$，使得 $t = t_i$，或者确定不存在这样的 t_i*。明显而自然的方法就是把 t 分别与 t_1 相比，与 t_2 相比，与 t_3 相比，等等，直到找到一个匹配的数或者到达数表的末尾。平均这需要进行 $n/2$ 次比较。如果我们对数表的状态没有做出任何假设，则通常我们所做的比较几乎等于 $n/2$。这就是**线性搜索**。

二分法搜索。如果数表 $L = \{s_1, s_2, \cdots, s_n\}$ 已经被排为升序，可以大提高我们的搜索速度：给定另一个元素 t，我们找到 $t_i \in L$，使得 $t = t_i$，或者确定不存在这样的 t_i，我们将最多使用 ceiling($\log_2 n$)（取整函数，恰好超过 $\log_2 n$ 的整数）次 t 与表中元素的比较。这种方法采用了一种*分而治之*的策略。为简便起见，我们假设 $n = 2^k$，k 为一个整数（如若不然，我们可以在表的末尾添加无意义的元素使其长度为 2 的方幂）。首先将 t 与 $t_{n/2}$ 比较，如果 $t \leqslant t_{n/2}$，则我们将在长为 $n/2$ 的表 $\{s_1, s_2, \cdots, s_{n/2}\}$ 上继续搜索；如果 $t > t_{n/2}$，则我们将在长为 $n/2$ 的表 $\{s_{\frac{n}{2}+1}, \cdots, s_n\}$ 上继续搜索。这是一次比较。对长度做归纳法，每一个这样的对分搜索最多需要 $\log_2 \frac{n}{2} = (\log_2 n) - 1$ 次比较。因此，由归纳法，递归地搜索我们最多需要 $\log_2 n$ 次比较。如果 n 不是 2 的方幂，则存在一个微小的差异，所以它需要 ceiling($\log_2 n$) 次比较。这就是**二分法搜索**。

备注 有序表上的二分法搜索与无序表上的线性搜索速度之比较，清楚地表明了针对有序表的快速算法的重要性。

最大数和最小数。为了寻找一个无序整数表 $\{t_1, t_2, \cdots, t_n\}$ 中的最大数或最小数，需要 $n-1$ 次比较。一种寻找最小数的进行了 $n-1$ 比较的方法概述如下：第一步取 $b = t_1$ 作为候选的最

小数。将 $b=t_1$ 与 t_2 比较，如果 $t_2<t_1$，则取 $b=t_2$ 作为新的候选最小数，否则仍然保留 $b=t_1$ 为候选最小数。将 b 与 t_3 比较，如果 $t_3<b$，则取 $b=t_3$ 作为新的候选最小数，否则仍然保留 b 不变。继续执行比较操作，直到完成了与所有 t_i 的比较。最后得到的 b 就是最小数。

插入排序是一种玩纸牌中常用的排序方法。该方法对一般的数据速度较慢。给定一个无序的数表 $L=\{t_1,\cdots,t_n\}$，一个新的表 $M=\{s_1,\cdots,s_n\}$ 按照如下方法逐步产生：取 s_1 为表 L 中的最小数，取 s_2 为表 L 中去掉 s_1 后的最小数，取 s_3 为表 L 中去掉 s_1 和 s_2 后的最小数。反复进行这一步骤，取 s_{n-1} 为表 L 中去掉 $s_1,\cdots,s_{n-2}$ 后的最小数，最后一个剩余的元素就是 s_n。这总共需要 $(n-1)+(n-2)+\cdots+3+2+1=n(n-1)/2=O(n^2)$ 次比较。

备注 注意在插入排序中，我们反复舍弃由比较所获得的信息。尽管如何利用这种信息还不完全清楚，但反复舍弃它可能会使我们怀疑存在可能被纠正的效率低下问题。同样的低效率问题还出现在选择排序中。比如，堆排序是选择排序的一种更为精巧的形式，它通过创建和维护元素的堆结构来使用额外的信息。

选择排序与插入排序稍有不同。但需要相同的运行时间。为将一个正整数表 $L=\{t_1,t_2,\cdots,t_n\}$ 按照大小升序排列，选择排序将按如下方式工作。首先，找到表中的最小整数 t_{i_1}，然后把它移到表的开头，其他元素向后移位。第二，找出第二最小的整数 t_{i2}，将其移到第二个位置，其余元素向后移位。第三，找出第三最小的整数 t_{i3}，将其移到第三个位置，其余元素向后移位。以此类推，直到排好序，这需要 $O(n^2)$ 次比较。

冒泡排序是一个比较聪明的排序方法：反复交换顺序不正确的邻近元素，直到整个表的顺序排好。该方法速度也较慢，在较坏的情况下，对长为 n 的表排序共需要 $O(n^2)$ 操作。另一方面，如果数据几乎接近排好序，则冒泡排序法会很快完成排序。因此，在一些比较成熟的排序算法（混合排序）中，冒泡排序法可以作为其中的一部分。

合并排序较上述排序法速度要快一些，而且可在 $O(n\log n)$ 时间内完成，而不是 $O(n^2)$。它也使用分而治之的策略。为简便起见，假设需要排序的数表有 2^l 个元素（如若不然，我们可以在表的末尾添加无意义的元素使其长度为 2 的方幂）。合并排序法有如下递归的定义：将整个数表分成两部分，分别对每部分合并排序，然后再合并。合并操作应当将两个长度为 2^{k-1} 且排好序的数表合并成一个长度为 $n=2^k$ 的有序表。完成将两个有序的数表 $L_1=(s_1,\cdots,s_a)$ 和 $L_2=(t_1,\cdots,t_b)$ 合并的操作至多需要

$$a+b-1=L_1\text{的长度}+L_2\text{的长度}-1$$

次比较，操作过程如下。首先，如果 $s_1<t_1$，则取 s_1 作为合并后数表的第一个元素，然后将它从 L_1 中去掉；否则，取 t_1 作为合并后数表的第一个元素，然后将它从 L_2 中去掉。无论在那种情况，新的 L_1 和 L_2 的长度之和减少 1，而且做了一次比较。由对 $a+b$ 的归纳，就证明了完成合并需要至多 $a+b-1$ 次比较。

经过反复的细分之后，合并排序则会面临这样一个问题：要对 n 个单独的包含单个元素的表排序。每个这样的表根本不需要比较。那么合并排序需要对 $n/2$ 对单元素数表进行合并。每个这样的合并至多需要 $1+1-1$ 次比较，所以共需要 $(n/2)(1+1-1)$ 次比较。那么对 $n/4$ 对含两个元素的表进行合并则至多需要 $(n/4)(2+2-1)$ 次比较。为了使计算简单，我们忽略所有这些“-1”，因为我们可以认为合并两个长度分别为 a 和 b 的有序表至多需要 $a+b$ 次比较。这样合并操作将总共需要至多

$$\frac{n}{2}\cdot 2+\frac{n}{4}\cdot 4+\frac{n}{8}\cdot 8+\cdots+8\cdot\frac{n}{8}+4\cdot\frac{n}{4}+2\cdot\frac{n}{2}\leqslant n\cdot\log_2 n$$

次比较。这也是我们对合并排序所期望的估计值。对于较大的n，这个估计值比那些天真排序算法所需要的$n^2/2$次的比较要小。

备注 合并排序的一个不足之处是即使是上述粗略描述的简单实现，也需要额外的存储器，其数量等于原始表的长度。

备注 注意在前面的所有算法中我们忽略了对表中元素移位的成本。从一定程度上讲这对上面指定的算法是适当的，因为在整个工作量中比较操作是最关键的一部分。

快速排序是另外一种分而治之的排序算法。我们将首先描述它的原始形式，其性能是很差的，然后我们将对其进行修改以避免最坏情况并利用其最优情况的动作。那么这样的话它的运行速度就与合并排序相当，如果实现得好的话，不需要过多的存储器。快速排序的这种改进形式已成为一些应用中的排序工具，如 Unix/Linux，Perl 的内置排序，*C* 的 qsort,Python 调用 *C* 的 qsort。它本质上是概率意义上的算法，这一点有别于其他算法。

给定一个需要排序的表L，快速排序法首先猜测整个表的中点。这个被猜测的值称为**中心点**（pivot）。然后将表划分为两部分，即中心点以下的元素和中心点以上的元素。这样的每一部分再进行快速排序。因此该算法的最简单形式递归地调用自己，同合并排序一样。

对中心点的最简单的选择就是选取表中的最后一个元素或者第一个元素。随机选取也是合理的，但是需要生成一个随机数。前提条件就是两部分的大小要相当。如果是这样，则运行时间的分析大体上与合并排序的相当，即$O(n\log n)$。但是，如果中心点的选择很差，使得表的一部分与另一部分相比显得很小，则运行时间会变为$O(n^2)$。

解决差中心点问题的方法就是采用**中值法**选择中心点。该方法就选择表上的三个“随机”元素，并以中间值作为中心点。概率计算表明这样选择中心点就很可能产生大小相当的划分，因此从统计意义上保证了好的运行时间。

备注 当用程序实现快速排序和合并排序时，要尽可能避免让操作系统实现算法递归地调用自己，要明确地处理好递归调用。这可能提高速度并避免因为递归（循环）太深而产生的警告。

A.3 向量

首先我们回忆一下，在解析几何中，一个有序的实数对(x,y)通常用来表示平面上的一个点，而一个有序的 3 元组(x,y,z)则表示三维空间中的一个点。尽管我们可以很容易地写出一个 7 元组，但却无法看清楚这到底是一个 7 维的什么东西。实际上这也不是一个问题。

同时，在**向量**的第一种形式中我们可以看到，一个n维向量就是一个有序的n元组（实数或者复数）$v=(v_1,\cdots,v_n)$。

那么如何区别“向量”与“点”呢？这也不是一个问题：使用向量来表示一个点就是向量的概念在物质世界的具体解释的一个例子。

维数相同的两个向量x和y有一个**向量加法**定义为：

$$x+y=(x_1,\cdots,x_n)+(y_1,\cdots,y_n)=(x_1+y_1,x_2+y_2,\cdots,x_n+y_n)$$

在这里一个**标量**就是一个简单的数。注意对标量加法同样使用向量法的+符号，并且没有提及向量的维数（这里也不必要在变量上加一个箭头来表示向量）。n维**零向量**为$0=(0,0,\cdots,0)$。

（同样的符号0既表示任何维数的向量，也表示标量，上下文应当能够看出来它到底指什么）

标量乘以向量的**标量乘法**定义如下：对标量 c 和向量 $x=(x_1,\cdots,x_n)$，

$$cx=c(x_1,x_2,\cdots,x_n)=(cx_1,cx_2,\cdots,cx_n)$$

而且可以记 $-x=(-x_1,-x_2,\cdots,-x_n)$，对 $y=(y_1,\cdots,y_n)$

$$x-y=(x_1-y_1,x_2-y_2,\cdots,x_n-y_n)$$

由毕达哥拉斯定理，我们可以得到二维空间的距离公式：

$$(x_1,x_2)\text{到}(y_1,y_2)\text{的距离}=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2}$$

类似地我们还有三维空间的距离公式：

$$(x_1,x_2,x_3)\text{到}(y_1,y_2,y_3)\text{的距离}=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+(x_3-y_3)^2}$$

一般地，对 n 维向量 $x=(x_1,\cdots,x_n)$，x 的**范数**或**长度**为 $|x|=\sqrt{x_1^2+x_2^2+\cdots+x_n^2}$，并且由一个 n 维向量 $x=(x_1,\cdots,x_n)$（它所表示的点）到另一个 n 维向量 $y=(y_1,\cdots,y_n)$（它所表示的点）的距离为：

$$|x-y|=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2}$$

作为一种特殊情况，$|x|=|x-0|$，那么 x 的范数就与点 x 到 0 的距离相同。

两个 n 维向量 $x=(x_1,\cdots,x_n)$ 和 $y=(y_1,\cdots,y_n)$ 的**内积**、**标量积**或者**点积**为：

$$x_1y_1+x_2y_2+x_3y_3+\cdots+x_ny_n=x\cdot y=\langle x,y\rangle$$

注意向量点积的值是标量，而不是向量。

命题 设 x,y,z 为 n 维向量，c 为一个标量，则有如下结论：

- $c(x+y)=cx+cy$
- $x+(y+z)=(x+y)+z$
- $x\cdot y=y\cdot x$
- $x\cdot(y+z)=x\cdot y+x\cdot z$ 且 $(y+z)\cdot x=y\cdot x+z\cdot x$
- $(cx)\cdot y=c(x\cdot y)$
- $|x|=\sqrt{x\cdot x}$

也许你可能感到吃惊，点积可以用范数来表示（即用距离来表示）：

命题 （极化恒等式）对两个 n 维向量 x 和 y，如下等式成立：

$$\langle x,y\rangle=\frac{1}{4}(|x+y|^2-|x-y|^2)$$

证明 如果我们将等式左右两边的向量展开成分量形式，也不难证明恒等式。但我们这里利用前面的性质如点积的分配律、交换律及距离表示来证明

$$\begin{aligned}&|x+y|^2-|x-y|^2=\langle x+y,x+y\rangle-\langle x-y,x-y\rangle\\&=\langle x,x\rangle+\langle x,y\rangle+\langle y,x\rangle+\langle y,y\rangle-\langle x,x\rangle+\langle x,y\rangle+\langle y,x\rangle-\langle y,y\rangle\\&=2\langle x,y\rangle+2\langle y,x\rangle=4\langle x,y\rangle\end{aligned}$$

结论得证。 ♣

关于标量/点积的一个最显然也是最常用的性质由下述定理给出：

定理 设 x 和 y 为两个 n 维向量，且都不为零。令 S_x 是由原点到 x 点的线段，S_y 是由原点到 y 点的线段，并设 θ 是 S_x 和 S_y 之间的夹角，则

$$\cos\theta = \frac{\langle x, y\rangle}{|x\|y|}$$

证明 我们将问题简化为一个二维向量的三角问题。我们将使用一个类似于极化恒等式的结论：$\langle x, y\rangle = \frac{1}{2}(|x|^2 + |y|^2 - |x-y|^2)$。这样定理的结论就变为：

$$\cos\theta = \frac{|x|^2 + |y|^2 - |x-y|^2}{2|x\|y|}$$

上式右边只涉及 x 与 y 以及 x 和 y 与 0 之间的距离。

这里的三个点 x、y 和 0 在同一个平面，无论空间的维数多大，它们仍然在同一平面。这三个点就是平面上一个三角形 T 的顶点。θ 就是三角形 T 在顶点 0 的内角。因此，如果等式 $\cos\theta = \frac{|x|^2 + |y|^2 - |x-y|^2}{2|x\|y|}$ 对二维向量成立，那么它对任意维的向量都成立。二维向量的情况作为练习。 ♣

定理 （*Cauchy-Schwarz-Bunyakowsky* 不等式）设 x 和 y 为两个 n 维向量，则下述不等式成立：$\langle x, y\rangle \leqslant |x\|y|$。等号成立，如果一个向量是另一个向量的倍数。

证明 这个结论实际上是两个向量之间夹角作余弦公式的一个直接结果，但在这里我们将不依靠任何几何的假设，给出另一种证明。

设 t 是一个标量，并考虑函数 $f(t) = |x+ty|^2 = |x|^2 + 2t\langle x, y\rangle + t^2|y|^2$，该函数右边的展开式利用了前面的性质。当然一个向量的范数是非负的，并且只有当向量为 0 时它才为 0。因此 $f(t)$ 对所有 t 都是非零的，除非 x 是 y 的倍数。

另一方面，我们还可以假设 y 不为零向量，否则定理的结论就变为简单的两边为 0 的情况。现在我们来求 $f(t)$ 的最小值，对 $f(t)$ 求 t 的导数，令导数为零以求解极值点：

$$0 = 2\langle x, y\rangle + 2t|y|^2$$

因此 $t = \frac{-\langle x, y\rangle}{|y|^2}$。（我们知道，这个关键点偶然可能为**最大值**，而非最小值，但实际上我们并不对此作任何假设！）当然 $f(t)$ 在这一点的值是非负的，将 $t = \frac{-\langle x, y\rangle}{|y|^2}$ 代入 $f(t)$ 即得：

$$0 \leqslant f\left(\frac{-\langle x, y\rangle}{|y|^2}\right) = |x|^2 - \frac{\langle x, y\rangle^2}{|y|^2}$$

整理后即得 $\langle x, y\rangle^2 \leqslant |x|^2|y|^2$。由此即得定理的结论。 ♣

习题

A.3.01 对维数相同的三个向量 x, y, z，证明 $x\cdot(y+z) = x\cdot y + x\cdot z$。

A.3.02 证明对标量 c 和向量 x，$|cx| = |c\|x|$。

A.3.03 证明 $|x-y|^2 = |x|^2 - 2\langle x, y\rangle + |y|^2$。

A.3.04 证明 $\langle x, y\rangle = \frac{1}{2}(|x|^2 + |y|^2 - |x-y|^2)$。

A.3.05 **菱形**的形状像钻石，也就是说，它是平面上四个边等长的多边形，因此对边是平行的。证明（利用向量几何！？）：对角线的平方和等于周长。

A.3.06 在二维向量空间，三角形的三个顶点为 A、B、C，顶点 C 的顶角 θ 的余弦值为

$$\cos\theta=\frac{|AC|^2+|BC|^2-|AB|^2}{2|AC||BC|}$$

这里的 $|AB|$ 就点 A 到点 B 之间线段的长度。（**余弦定理**的一个版本。）

A.3.07 证明 $(1+2+3+\cdots+n)^2\leqslant n\cdot(1^2+2^2+\cdots+n^2)$。

A.4 矩阵

一个 $m\times n$ **矩阵**就是一个的数块

$$x=\begin{pmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2n} \\ x_{31} & x_{32} & x_{33} & \cdots & x_{3n} \\ & & \cdots & & \\ x_{m1} & x_{m2} & x_{m3} & \cdots & x_{mn} \end{pmatrix}$$

处于第 i 行第 j 列的元素称为第 (i,j) 个**元素**。如果 x 是一个矩阵，则它的第 (i,j) 个元素记为 x_{ij}。

通常我们把 $n\times 1$ 矩阵称为 n 维**列向量**，而把 $1\times n$ 矩阵称为 n 维**行向量**。

矩阵加法就是两个行列相同的矩阵对应元素直接相加：

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2n} \\ x_{31} & x_{32} & x_{33} & \cdots & x_{3n} \\ & & \cdots & & \\ x_{m1} & x_{m2} & x_{m3} & \cdots & x_{mn} \end{pmatrix}+\begin{pmatrix} y_{11} & y_{12} & y_{13} & \cdots & y_{1n} \\ y_{21} & y_{22} & y_{23} & \cdots & y_{2n} \\ y_{31} & y_{32} & y_{33} & \cdots & y_{3n} \\ & & \cdots & & \\ y_{m1} & y_{m2} & y_{m3} & \cdots & y_{mn} \end{pmatrix}$$

$$=\begin{pmatrix} x_{11}+y_{11} & x_{12}+y_{12} & x_{13}+y_{13} & \cdots & x_{1n}+y_{1n} \\ x_{21}+y_{21} & x_{22}+y_{22} & x_{23}+y_{23} & \cdots & x_{2n}+y_{2n} \\ x_{31}+y_{31} & x_{32}+y_{32} & x_{33}+y_{33} & \cdots & x_{3n}+y_{3n} \\ & & \cdots & & \\ x_{m1}+y_{m1} & x_{m2}+y_{m2} & x_{m3}+y_{m3} & \cdots & x_{mn}+y_{mn} \end{pmatrix}$$

$m\times n$ 的**单位元**或者**零矩阵** $0_{m,n}$ 就是一个所有元素都为 0 的 m,n 矩阵。它显然的性质就是它与任何与它行列相同的矩阵 x 相加仍得：

$$x+0_{m,n}=x=0_{m,n}+x$$

当然还有**乘法矩阵**，它的含义和计算比较复杂：对一个 $k\times m$ 矩阵 x 和一个 $m\times n$ 矩阵 y，乘积 xy 的第 (i,j) 个元素不仅依赖于 x 的第 i 行，还依赖于 y 的第 j 列：

$$\begin{pmatrix} & \cdots & \\ \cdots & (xy)_{ij} & \cdots \\ & \cdots & \end{pmatrix} = \begin{pmatrix} & \cdots & \\ x_{i1} & x_{i2} & x_{i3} \quad \cdots \quad x_{im} \\ & \cdots & \end{pmatrix} \begin{pmatrix} \cdots & y_{1j} & \cdots \\ \cdots & y_{2j} & \cdots \\ \cdots & y_{3j} & \cdots \\ \cdots & \vdots & \cdots \\ \cdots & y_{mj} & \cdots \end{pmatrix}$$

$$= \begin{pmatrix} & \cdots & \\ \cdots & x_{i1}y_{1j} + x_{i2}y_{2j} + \cdots + x_{im}y_{mj} & \cdots \\ & \cdots & \end{pmatrix}$$

即乘积的第 (i, j) 个元素为：

$$(xy)_{ij} = x_{i1}y_{1j} + x_{i2}y_{2j} + \cdots + x_{im}y_{mj}$$

$$= \sum\nolimits_{1 \le l \le m} x_{il}y_{lj} = \sum\nolimits_{l=1}^{m} x_{il}y_{lj}$$

$n \times n$ 阶**单位矩阵** 1_n 就是**对角线**上元素为1而其他元素全为0的矩阵，即

$$1_n = \begin{pmatrix} 1 & 0 & 0 & \cdots & \\ 0 & 1 & 0 & \cdots & \\ 0 & 0 & 1 & \cdots & \\ & & & \cdots & \\ & & & 1 & 0 \\ & & \cdots & 0 & 1 \end{pmatrix}$$

单位矩阵 1_n 具有这样的性质，即对任何的 $m \times n$ 矩阵 x 和一个 $n \times m$ 矩阵 y，有如下等式：

$$x \cdot 1_n = x, \quad 1_n \cdot y = y$$

因为一个 $k \times m$ 矩阵和一个 $m \times n$ 矩阵的乘积为 $k \times n$ 矩阵，所以对任意正整数 n，$n \times n$ 阶方阵既有乘法也有加法，而且结果仍为 $n \times n$ 阶方阵。

有些 $n \times n$ 阶方阵 x 存在**乘法逆矩阵** x^{-1}，即 $x \cdot x^{-1} = 1_n = x^{-1} \cdot x$。这样的方阵称为是**可逆的**。一般地，对较大的矩阵，寻找其逆矩阵（假设存在一个）的过程从计算上是较精密的。

注意在上述的运算中，根本没有依赖向量和矩阵中所用数的类型。特别地，如果我们使用取自 $\mathbf{Z}_n$ 的元素(不提实数、复数或有理数元素)，则所有运算都是有意义的。

命题 矩阵乘法满足结合律。即对三个行列相当的矩阵 A, B, C，$A(BC) = (AB)C$。

证明 设 A_{ij}, B_{ij}, C_{ij} 分别是 A, B, C 的第 (i, j) 个元素，令 $(AB)_{ij}$ 是 AB 的第 (i, j) 个元素，等等。为了证明两个矩阵的乘积相等，需要我们证明两个乘积的对应项相等，利用前面矩阵乘积的公式，有：

$$(A(BC))_{ij} = \sum_{l} A_{il}(BC)_{lj} = \sum_{l} A_{il} \sum_{k} B_{lk}C_{kj}$$

$$= \sum_{l,k} A_{il}B_{lk}C_{kj} = \sum_{k} (\sum_{l} A_{il}B_{lk})C_{kj}$$

$$= \sum_k (AB)_{ik} C_{kj} = ((AB)C)_{ij}$$

因此，结合律成立。 ♣

设 R 为 $\mathbf{Q}$、$\mathbf{R}$、$\mathbf{C}$ 或 $\mathbf{Z}_n$，则所有取值于 R 的 $n\times n$ 阶可逆矩阵构成的集合记为 $GL(n,R)$ 或者 $GL_n(R)$，称为**一般线性群**。

习题

A.4.01 证明矩阵 $\begin{pmatrix}1 & 3\\ 2 & 6\end{pmatrix}$ 不属于 $GL(2,\mathbf{R})$，即它没有逆矩阵。

A.4.02 找出两个乘法不交换的 2×2 矩阵 A,B，即满足 $AB\neq BA$。

A.4.03 找出两个 2×2 矩阵 A,B，它们都不为零矩阵，但它们的乘积为零矩阵。

A.4.04 证明对任意的正整数 N，$\begin{pmatrix}1 & 1\\ 0 & 1\end{pmatrix}^N = \begin{pmatrix}1 & N\\ 0 & 1\end{pmatrix}$。

A.4.05 设矩阵 $x=\begin{pmatrix}1 & 1 & 0\\ 0 & 1 & 1\\ 0 & 0 & 1\end{pmatrix}$，对正整数 N，确定计算 x^N 的公式，并证明它的正确性。

A.4.06 设 A,B 是两个 $n\times n$ 可逆矩阵，证明 $(AB)^{-1}=B^{-1}A^{-1}$。

A.4.07 在 $GL(2,\mathbf{Z}/2)$ 中有多少个元素？（即在 $2^4=16$ 个取值于 $\mathbf{Z}/2$ 的 2×2 矩阵中有多少个可逆矩阵）

A.4.08 **斐波那契数**的递归定义如下：$F_0=0$，$F_1=1$ 且 $F_{n+1}=F_{n-1}+F_n$。令 $M=\begin{pmatrix}0 & 1\\ 1 & 1\end{pmatrix}$，证明对任意正整数 n，$\begin{pmatrix}F_{n+1}\\ F_n\end{pmatrix}=M^n\begin{pmatrix}1\\ 0\end{pmatrix}$。

A.4.09 设 F_n 是第 n 个斐波那契数，令 $M=\begin{pmatrix}0 & 1\\ 1 & 1\end{pmatrix}$，证明 $M^n=\begin{pmatrix}F_n & F_{n+1}\\ F_{n+1} & F_{n+2}\end{pmatrix}$。

A.4.10 证明对任何形如 $M=\begin{pmatrix}0 & x & y\\ 0 & 0 & z\\ 0 & 0 & 0\end{pmatrix}$ 的矩阵有 $M^3=0_3$。

A.4.11 设 $M=\begin{pmatrix}0 & x & y\\ 0 & 0 & z\\ 0 & 0 & 0\end{pmatrix}$，证明 1_3-M 可逆且其逆矩阵 $(1_3-M)^{-1}=1_3+M+M^2$。

A.4.12 一个矩阵 M 称为是**严格三角矩阵**，如果对 $i\geqslant j$，$M_{ij}=0$。证明一个 $n\times n$ 上三角矩阵 M 满足 $M^n=0_n$。

A.4.13 设 M 是一个 $n\times n$ 阶的严格三角矩阵，证明 1_n-M 是一个可逆矩阵且其逆矩阵 $(1_n-M)^{-1}=1_n+M+M^2+M^3+\cdots+M^{n-1}$。

A.5 斯特林公式

阶乘函数 $n!=1\cdot 2\cdot 3\cdot 4\cdot 5\cdot\ \cdots\ \cdot(n-2)\cdot(n-1)\cdot n$ 的性质对我们理解和掌握那些答案涉及阶

乘的计数问题具有非常重要的作用。

阶乘函数 $n!$ 最明显的特征就是它为一个以 n 为变量的增长非常快速的函数。因此，在许多情况下，问题可能不是精确地计算阶乘，而是用一个比较简单且我们容易计算其大小的函数来表示它们。我们这里得到的是**斯特林估计**的最简单情形，但已经足够了。这个结论本身的证明也是值得关注的。

（这个估计在素数定理的基本方法中作用也很明显。）

命题 极限 $\lim\limits_{n\to\infty}\dfrac{n!}{n^{n+\frac{1}{2}}e^{-n}}$ 存在，且值为 $\sqrt{2\pi}$。此外，对于 $n\geqslant 2$

$$\sqrt{2\pi}\,n^{n+\frac{1}{2}}e^{-n}\cdot e^{\frac{1}{12(n+1)}}<n!<\sqrt{2\pi}\,n^{n+\frac{1}{2}}e^{-n}\cdot e^{\frac{1}{12n}}$$

证明 我们的证明以作如下比较开始。首先比较

$$\ln(n!)=\ln 1+\ln 2+\ln 3+\cdots+\ln n$$

与积分

$$\int_1^{n+1}\ln x\,dx=[x\ln x-x]_1^{n+1}=(n+1)\ln(n+1)-(n+1)$$

以及

$$\int_0^{n}\ln x\,dx=[x\ln x-x]_0^{n}=n\ln n-n$$

（后一个整数不等于 0，但趋近于 0。）因为 ln 函数是一个增函数，所以对 $n=2,3,4,\cdots$，$\int_{t-1}^{t}\ln x\,dx\leqslant\ln t\leqslant\int_t^{t+1}\ln x\,dx$

由此我们得 $n\ln n-n\leqslant\ln n!\leqslant(n+1)\ln(n+1)-(n+1)$，更进一步，我们得到

$$n^ne^{-n}\leqslant n!\leqslant(n+1)^{n+1}e^{-(n+1)}$$

这里我们选择了一个很关键的量 $(n+\frac{1}{2})\ln n-n$，它是介于我们刚才得到的上界和下界之间一个平均值。令 $E_n=\ln n!-[(n+\frac{1}{2})\ln n-n]$ 是用这个平均值来估计 $\ln n!$ 时的误差。由此有：

$$\begin{aligned}E_n-E_{n+1}&=\ln n!-[(n+\tfrac{1}{2})\ln n-n]-[\ln(n+1)!-[(n+1+\tfrac{1}{2})\ln(n+1)-(n+1)]]\\&=(n+\tfrac{1}{2})\ln(1+\tfrac{1}{n})-1\end{aligned}$$

利用泰勒展开式 $\ln(1+x)=x-\dfrac{x^2}{2}+\dfrac{x^3}{3}-\dfrac{x^4}{4}+\cdots$，可以获得

$$\begin{aligned}E_n-E_{n+1}&=\left(n+\frac{1}{2}\right)\left(\frac{1}{n}-\frac{1}{2n^2}+\frac{1}{3n^3}-\cdots\right)-1\\&=\left(\frac{1}{1}-\frac{1}{2n}+\frac{1}{3n^2}-\cdots\right)+\frac{1}{2}\left(\frac{1}{n}-\frac{1}{2n^2}+\frac{1}{3n^3}-\cdots\right)-1\\&=\left(\frac{1}{3}-\frac{1}{4}\right)\frac{1}{n^2}+\left(-\frac{1}{4}+\frac{1}{6}\right)\frac{1}{n^3}+\left(\frac{1}{5}-\frac{1}{8}\right)\frac{1}{n^4}+\left(-\frac{1}{6}+\frac{1}{10}\right)\frac{1}{n^5}+\cdots\end{aligned}$$

注意对于 $n\geqslant 1$，这是一个交错下降序列。

回忆一下交错下降序列的不等式：对一个交错序列 $a_1-a_2+a_3-a_4+\cdots$（即对所有 i，$a_i>0$ 且 $a_i>a_{i+1}$），有如下不等式成立：

$$a_1-a_2<a_1-a_2+a_3-a_4+\cdots<a_1-a_2+a_3$$

因此，$\dfrac{1}{12}\left(\dfrac{1}{n^2}-\dfrac{1}{n^3}\right)<E_n-E_{n+1}<\dfrac{1}{12n^2}-\dfrac{1}{12n^3}+\dfrac{1}{40n^4}$。

特别地，因为不等式的左边总是大于零，$E_n - E_{n+1}$ 的每个值都是正值，所以序列 E_n 本身是下降的。

给右半边不等式减去 $\frac{1}{12n}$ 并加上 $\frac{1}{12(n+1)}$，我们得到：

$$(E_n - \frac{1}{12n}) - (E_{n+1} - \frac{1}{12(n+1)} < \frac{1}{12n^2} - \frac{1}{12n^3} + \frac{1}{40n^4} - \frac{1}{12n} + \frac{1}{12(n+1)}$$

上面不等式的右边可以简化为 $\frac{12-28n}{12\cdot 40\cdot n^3(n+1)}$，对所有 $n\geqslant 1$，这个值是负值，因此序列 $E_n - \frac{1}{12n}$ 是上升的。

因为 E_n 本身是下降的，而 $E_n - \frac{1}{12n}$ 却是上升的，并且 $\frac{1}{12n}$ 是趋于零的，因此我们断定 E_n 是有界的下降序列，所以它有极限 C。实际上这个极限为 $\sqrt{2\pi}$，这里就不再证明了。

同样，给左半边不等式减去 $\frac{1}{12(n+1)}$ 并加上 $\frac{1}{12((n+1)+1)}$，我们得到：

$$\frac{1}{12}(\frac{1}{n^2} - \frac{1}{n^3}) - \frac{1}{12(n+1)} + \frac{1}{12((n+1)+1)} < (E_n - \frac{1}{12(n+1)}) - (E_{n+1} - \frac{1}{12((n+1)+1)})$$

对左边的式子进行简化（不如前面那么不可思议）合并后得到：$\frac{2n^2-n-2}{n^3\cdot 12(n+1)\cdot 12((n+1)+1)}$ 对 $n\geqslant 2$，该式的分子部分满足

$$2n^2 - n - 2 = 2\cdot[(n-\frac{1}{4})^2 - \frac{1}{16} - 1] \geqslant 2\cdot[(2-\frac{1}{4})^2 - \frac{17}{16}] = 4 > 0$$

因此，至少从 $n\geqslant 2$ 开始，序列 $E_n - \frac{1}{12(n+1)}$ 是下降的。

总之，$\lim_n E_n = C$，并且因为序列 $\frac{1}{12n}$ 和 $\frac{1}{12(n+1)}$ 都趋于零，因此 $E_n - \frac{1}{12n}$ 上升地趋于 C，而 $E_n - \frac{1}{12(n+1)}$ 下降在趋于 C。

因此对于 $n\geqslant 2$，$C + \frac{1}{12(n+1)} < E_n < C + \frac{1}{12n}$。这也就是有如下不等式：

$$C + \frac{1}{12(n+1)} + (n+\frac{1}{2})\ln n - n < \ln n! < C + \frac{1}{12n} + (n+\frac{1}{2})\ln n - n$$

这就证明了命题的结论。 ♣

习题

A.5.01 利用斯特林公式，证明存在一个常数 C 使得我们可以用下式估计二项式展开“中间项”的系数：$\binom{2n}{n} \leqslant C\cdot 2^{2n}\cdot\frac{1}{\sqrt{n}}$。

A.5.02 证明在 $2n$ 次抛掷硬币的试验中，恰好有 n 次正面朝上的概率小于 $C/\sqrt{n}$，C 为某个常数。

A.5.03 设 $a_1 > a_2 > a_3 > \cdots$ 为一个正实数序列，且 $\lim_n a_n = 0$。令

$$S_n = a_1 - a_2 + a_3 - \cdots + (-1)^{n-1} a_n$$

为这个**交错下降和**的 n 次部分和。证明对所有的指标 n，$S_{2n} < S_{2n+2}$，$S_{2n-1} > S_{2n+1}$。（即偶项部分和是上升的，而奇项部分和是下降的）

A.5.04 设 $a_1 > a_2 > a_3 > \cdots$ 为一个正实数序列，且 $\lim_n a_n = 0$。假设

$$L = a_1 - a_2 + a_3 - a_4 + \cdots$$

的极限存在。证明对所有的指标 k，如下不等式成立：

$$a_1 - a_2 + a_3 - a_4 + \cdots + a_{2k-1} - a_{2k} < L < a_1 - a_2 + a_3 - a_4 + \cdots + a_{2k-1}$$

A.5.05(*) 证明一个交错下降的无限和是收敛的。即 S_n 为 n 次部分和，同 **A.5.03**。证明对每个 $\varepsilon > 0$，存在一个正整数 N，使得对所有 $i, j \geqslant N$，$|S_i - S_j| < \varepsilon$。

附录B 部分习题答案

1.1.01 'GUVF VF GUR ZRFFNTR'

1.1.02 'AOPZ PZ AOL TLZZHNL'

1.1.07 'Take me out to the ball game'

1.1.08 'Our State fair is the best'

1.1.09 'Oh, what a beautiful morning'

1.1.11 'Shark has pretty teeth'

1.2.01 10

1.2.02 29

1.2.03 10

1.2.06 78

1.2.07 44

1.2.08 122

1.2.09 56

1.2.18 77

1.2.20 69

1.3.01 'RBXICEWFV'

1.3.02 'VHDSJKBTV'

1.3.05 'DBZ LDW QFZIS'

1.4.01 'RTTM RT HM RFQUFZCM'

1.4.02 'DTTC DT BC DLIOLPAC'

1.4.06 (9,15)

1.4.08 (15,7)

1.4.12 $(a,b)=(2,25)$

1.4.13 $(a,b)=(9,18)$

1.4.15 $(a,b)=(7,9)$

1.4.16 $(a,b)=(17,10)$

2.1.01 $3!=6$

2.1.03 $5!=120$

2.1.06 $\binom{9}{3}=84$

2.1.08 $\binom{10}{5}=252$

2.2.02 $\binom{10}{5}/2^{10}\approx 0.24609$

2.2.04 $\left(\binom{8}{5}+\binom{8}{6}+\binom{8}{7}+1\right)/2^8$

2.2.07 $\frac{5}{5+6}\cdot\frac{5}{5+6}=\frac{25}{121}\approx 0.2066$

2.4.01 'Better late than never'

3.1.02 'Unicode is a relatively new idea for encoding larger alphabets and ideographic system by using two bytes per character'

3.1.03 'From the theorem just proven the ratio of primitive roots to all elements is most often above one quarter'

3.2.02 'theme'

3.2.04 'grimy'

3.2.08 'finally and bicycle'

3.3.02 (1 2 5)(3 4 7 6)，阶为 12

3.3.04 (1 6)(2 5 7 3 4)，阶为 10

3.3.05 $\begin{pmatrix}1&2&3&4&5&6&7\\5&1&7&2&7&6&3\end{pmatrix}$

3.3.07 $\begin{pmatrix}1&2&3&4&5&6&7\\5&1&7&2&6&4&3\end{pmatrix}$

3.3.09 $6\cdot5\cdot4\cdot3/4=90$

3.4.04 (1 2 4 8 3 6 12 11 9 5 10 7)

3.4.06 8 循环为(1 2 4 8 16 15 13 9)，(3 6 12 7 14 11 5 10)

3.5.03 使用 0,1,2,…,11：
(0) (11) (1 3 9 5 4) (2 6 7 10 8)

4.1.02 'BERW AV IC TUH OCLTY NIHVR BIQQWXHI'

4.1.04 'TSVK UJ QP AVV RTQMA HTKVZ RQFUWXYB'

4.2.02 3，90

4.2.06 1，1457

4.2.09 79

4.4.02 50/14

4.4.03 60/15

4.4.05 1/2

4.4.06 3

5.1.01 $n \cdot 2^{n-1}$

5.1.05 $x(1+x)/(1-x)^3$

5.2.02 35/6

5.3.02 由车贝雪夫不等式，概率小于等于 1/40

5.4.02 8

7.1.02 1,2,4,5,8,10,16,20,40,80

7.1.04 1,2,3,4,6,8,12,16,24,32,48,96

7.1.09 $10510100501 = 1 \cdot 100^5 + 5 \cdot 100^4 + 10 \cdot 100^3 + 10 \cdot 100^2 + 5 \cdot 100 + 1 = (100+1)^5$

7.2.03 $110111101111011 = 11011 \times 100000^2 + 11011 \times 100000 + 11011 = 11011(100000^2 + 100000 + 1)$，所以 11011 是一个真因子，通过试除法，3、7、11、13、31、37 也是。

7.3.02 347

7.3.05 $1 = 117 \times (-34) + 173 \times (23)$

7.3.07 $3 = 12345 \times 3617 + 54321 \times (-822)$

7.4.03 8

7.4.05 5

7.4.08 $(n-3)$

7.5.02 19

7.5.05 19

7.5.07 3239

7.7.05 8 个：1,5,7,11,13,17,19,23 模 24

7.7.08 8

7.7.10 1

7.7.15 1,12,131,142

7.8.02 $2^2=4, 2^3=8, 2^4\%11=5, 2^5\%11=10, 2^6\%11=9, 2^7\%11=7, 2^9\%11=3, 2^{10}\%11=1$

7.8.06 2

7.8.08 7

7.8.10 29

10.2.02 187183

10.2.05 9611

12.1.02 16

12.1.04 8

12.1.06 3

12.4.03 1,7,37,5×5×7, 11×71, 7×13×37, 14197, 5×5×7×337, 37×6553, 7×11×71×181, 23×174659, 5×5×7×13×37×193, 131×500, 111, 7×7×379×14,197, 11×37×61×71×601

12.5.02 334

12.5.04 559

12.5.06 17

12.5.08 18

12.6.02 5

12.6.04 60

12.6.07 928

12.7.02 59

12.7.03 31

12.7.05 19（还有 34、50）

13.2.02 1

13.2.04 89

13.2.06 385

13.3.05 1,103,118,220

13.4.02 315520,1456041

13.4.04 1068,14557

13.6.02 12,6,12

13.7.05 12

13.7.08 6

13.7.11 102

13.8.02 是

13.8.03 不是

13.8.06 不是

13.8.08 是

14.1.02 （3+1）×（1+1）=8

14.1.04 (1+3+9+27) ×(1+37)=1520

14.3.02 0

14.3.04 −1

15.5.02 +1，是

15.5.04 +1，是

15.5.09 +1，不是

15.5.10 +1，不是

16.2.03 25,325,561,703,817,1105

17.2.06 对于 $d=0,1,2,3,4,6,8,12$，d 的倍数模 24 的集合是一个子群，每一个这种倍数集合都是子群。

19.1.06 1,5,7,11 mod12

19.1.07 $3\cdot 7=0 \bmod 21$，
$6\cdot 7=0 \bmod 21, 9\cdot 7=0 \bmod 21$
$3\cdot 14=0 \bmod 21$ 等等。

19.2.02 $7\cdot 1=7\cdot 12 \bmod 77$，但
$1\neq 12 \bmod 77$

19.3.02 $5 \bmod 7$ 和 $6 \bmod 7$

19.3.04 $5 \bmod 13$ 和 $8 \bmod 13$

19.4.02 $x^8+x^7+x^6+x^2+x+1$

19.4.05 x^2+x+1

19.4.07 $x^8+x^6+x^3+1$

20.2.02 x^3+x^2+x+1

20.2.04 x^3+x+1

20.5.02 3,5,6,7,10,11,12,14

22.2.02 5,8

27.1.02 22

27.1.04 11

27.1.06 47

27.1.08 6

27.1.10 11

27.1.12 28

27.2.02 9

27.2.04 10

27.2.06 19

27.2.08 8

27.2.10 740

27.2.13 17

27.2.15 16

27.2.17 14

27.2.19 109

附录C 常用数表

在如今高性能的计算机普遍使用的时候，给出这样一些数表似乎有点傻。但是，在实际工作却表明，当用机器计算不可行时，有这样一个数表却是非常有用的。

表1 600以内的整数分解式

$2=2$	$3=3$	$4=2\cdot 2$	$5=5$
$6=2\cdot 3$	$7=7$	$8=2\cdot 2\cdot 2$	$9=3\cdot 3$
$10=2\cdot 5$	$11=11$	$12=2\cdot 2\cdot 3$	$13=13$
$14=2\cdot 7$	$15=3\cdot 5$	$16=2\cdot 2\cdot 2\cdot 2$	$17=17$
$18=2\cdot 3\cdot 3$	$19=19$	$20=2\cdot 2\cdot 5$	$21=3\cdot 7$
$22=2\cdot 11$	$23=23$	$24=2\cdot 2\cdot 2\cdot 3$	$25=5\cdot 5$
$26=2\cdot 13$	$27=3\cdot 3\cdot 3$	$28=2\cdot 2\cdot 7$	$29=29$
$30=2\cdot 3\cdot 5$	$31=31$	$32=2\cdot 2\cdot 2\cdot 2\cdot 2$	$33=3\cdot 11$
$34=2\cdot 17$	$35=5\cdot 7$	$36=2\cdot 2\cdot 3\cdot 3$	$37=37$
$38=2\cdot 19$	$39=3\cdot 13$	$40=2\cdot 2\cdot 2\cdot 5$	$41=41$
$42=2\cdot 3\cdot 7$	$43=43$	$44=2\cdot 2\cdot 11$	$45=3\cdot 3\cdot 5$
$46=2\cdot 23$	$47=47$	$48=2\cdot 2\cdot 2\cdot 2\cdot 3$	$49=7\cdot 7$
$50=2\cdot 5\cdot 5$	$51=3\cdot 17$	$52=2\cdot 2\cdot 13$	$53=53$
$54=2\cdot 3\cdot 3\cdot 3$	$55=5\cdot 11$	$56=2\cdot 2\cdot 2\cdot 7$	$57=3\cdot 19$
$58=2\cdot 29$	$59=59$	$60=2\cdot 2\cdot 3\cdot 5$	$61=61$
$62=2\cdot 31$	$63=3\cdot 3\cdot 7$	$64=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 2$	$65=5\cdot 13$
$66=2\cdot 3\cdot 11$	$67=67$	$68=2\cdot 2\cdot 17$	$69=3\cdot 23$
$70=2\cdot 5\cdot 7$	$71=71$	$72=2\cdot 2\cdot 2\cdot 3\cdot 3$	$73=73$
$74=2\cdot 37$	$75=3\cdot 5\cdot 5$	$76=2\cdot 2\cdot 19$	$77=7\cdot 11$
$78=2\cdot 3\cdot 13$	$79=79$	$80=2\cdot 2\cdot 2\cdot 2\cdot 5$	$81=3\cdot 3\cdot 3\cdot 3$
$82=2\cdot 41$	$83=83$	$84=2\cdot 2\cdot 3\cdot 7$	$85=5\cdot 17$
$86=2\cdot 43$	$87=3\cdot 29$	$88=2\cdot 2\cdot 2\cdot 11$	$89=89$
$90=2\cdot 3\cdot 3\cdot 5$	$91=7\cdot 13$	$92=2\cdot 2\cdot 23$	$93=3\cdot 31$
$94=2\cdot 47$	$95=5\cdot 19$	$96=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 3$	$97=97$
$98=2\cdot 7\cdot 7$	$99=3\cdot 3\cdot 11$	$100=2\cdot 2\cdot 5\cdot 5$	$101=101$
$102=2\cdot 3\cdot 17$	$103=103$	$104=2\cdot 2\cdot 2\cdot 13$	$105=3\cdot 5\cdot 7$
$106=2\cdot 53$	$107=107$	$108=2\cdot 2\cdot 3\cdot 3\cdot 3$	$109=109$
$110=2\cdot 5\cdot 11$	$111=3\cdot 37$	$112=2\cdot 2\cdot 2\cdot 2\cdot 7$	$113=113$
$114=2\cdot 3\cdot 19$	$115=5\cdot 23$	$116=2\cdot 2\cdot 29$	$117=3\cdot 3\cdot 13$

$118=2\cdot 59$ $119=7\cdot 17$ $120=2\cdot 2\cdot 2\cdot 3\cdot 5$ $121=11\cdot 11$

$122=2\cdot 61$ $123=3\cdot 41$ $124=2\cdot 2\cdot 31$ $125=5\cdot 5\cdot 5$

$126=2\cdot 3\cdot 3\cdot 7$ $127=127$ $128=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 2$ $129=3\cdot 43$

$130=2\cdot 5\cdot 13$ $131=131$ $132=2\cdot 2\cdot 3\cdot 11$ $133=7\cdot 19$

$134=2\cdot 67$ $135=3\cdot 3\cdot 3\cdot 5$ $136=2\cdot 2\cdot 2\cdot 17$ $137=137$

$138=2\cdot 3\cdot 23$ $139=139$ $140=2\cdot 2\cdot 5\cdot 7$ $141=3\cdot 47$

$142=2\cdot 71$ $143=11\cdot 13$ $144=2\cdot 2\cdot 2\cdot 2\cdot 3\cdot 3$ $145=5\cdot 29$

$146=2\cdot 73$ $147=3\cdot 7\cdot 7$ $148=2\cdot 2\cdot 37$ $149=149$

$150=2\cdot 3\cdot 5\cdot 5$ $151=151$ $152=2\cdot 2\cdot 2\cdot 19$ $153=3\cdot 3\cdot 17$

$154=2\cdot 7\cdot 11$ $155=5\cdot 31$ $156=2\cdot 2\cdot 3\cdot 13$ $157=157$

$158=2\cdot 79$ $159=3\cdot 53$ $160=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 5$ $161=7\cdot 23$

$162=2\cdot 3\cdot 3\cdot 3\cdot 3$ $163=163$ $164=2\cdot 2\cdot 41$ $165=3\cdot 5\cdot 11$

$166=2\cdot 83$ $167=167$ $168=2\cdot 2\cdot 2\cdot 3\cdot 7$ $169=13\cdot 13$

$170=2\cdot 5\cdot 17$ $171=3\cdot 3\cdot 19$ $172=2\cdot 2\cdot 43$ $173=173$

$174=2\cdot 3\cdot 29$ $175=5\cdot 5\cdot 7$ $176=2\cdot 2\cdot 2\cdot 2\cdot 11$ $177=3\cdot 59$

$178=2\cdot 89$ $179=179$ $180=2\cdot 2\cdot 3\cdot 3\cdot 5$ $181=181$

$182=2\cdot 7\cdot 13$ $183=3\cdot 61$ $184=2\cdot 2\cdot 2\cdot 23$ $185=5\cdot 37$

$186=2\cdot 3\cdot 31$ $187=11\cdot 17$ $188=2\cdot 2\cdot 47$ $189=3\cdot 3\cdot 3\cdot 7$

$190=2\cdot 5\cdot 19$ $191=191$ $192=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 3$ $193=193$

$194=2\cdot 97$ $195=3\cdot 5\cdot 13$ $196=2\cdot 2\cdot 7\cdot 7$ $197=197$

$198=2\cdot 3\cdot 3\cdot 11$ $199=199$ $200=2\cdot 2\cdot 2\cdot 5\cdot 5$ $201=3\cdot 67$

$202=2\cdot 101$ $203=7\cdot 29$ $204=2\cdot 2\cdot 3\cdot 17$ $205=5\cdot 41$

$206=2\cdot 103$ $207=3\cdot 3\cdot 23$ $208=2\cdot 2\cdot 2\cdot 2\cdot 13$ $209=11\cdot 19$

$210=2\cdot 3\cdot 5\cdot 7$ $211=211$ $212=2\cdot 2\cdot 53$ $213=3\cdot 71$

$214=2\cdot 107$ $215=5\cdot 43$ $216=2\cdot 2\cdot 2\cdot 3\cdot 3\cdot 3$ $217=7\cdot 31$

$218=2\cdot 109$ $219=3\cdot 73$ $220=2\cdot 2\cdot 5\cdot 11$ $221=13\cdot 17$

$222=2\cdot 3\cdot 37$ $223=223$ $224=2\cdot 2\cdot 2\cdot 2\cdot 2\cdot 7$ $225=3\cdot 3\cdot 5\cdot 5$

$226=2\cdot 113$ $227=227$ $228=2\cdot 2\cdot 3\cdot 19$ $229=229$

$230=2\cdot 5\cdot 23$ $231=3\cdot 7\cdot 11$ $232=2\cdot 2\cdot 2\cdot 29$ $233=233$

$234=2\cdot 3\cdot 3\cdot 13$ $235=5\cdot 47$ $236=2\cdot 2\cdot 59$ $237=3\cdot 79$

$238=2\cdot 7\cdot 17$ $239=239$ $240=2\cdot 2\cdot 2\cdot 2\cdot 3\cdot 5$ $241=241$

$242=2\cdot 11\cdot 11$ $243=3\cdot 3\cdot 3\cdot 3\cdot 3$ $244=2\cdot 2\cdot 61$ $245=5\cdot 7\cdot 7$

$246=2\cdot 3\cdot 41$ $247=13\cdot 19$ $248=2\cdot 2\cdot 2\cdot 31$ $249=3\cdot 83$

$250=2\cdot 5\cdot 5\cdot 5$ $251=251$ $252=2\cdot 2\cdot 3\cdot 3\cdot 7$ $253=11\cdot 23$

$254=2\cdot 127$ $255=3\cdot 5\cdot 17$ $256=2^8$ $257=257$

$258=2\cdot 3\cdot 43$ $259=7\cdot 37$ $260=2\cdot 2\cdot 5\cdot 13$ $261=3\cdot 3\cdot 29$

$262=2\cdot 131$ $263=263$ $264=2\cdot 2\cdot 2\cdot 3\cdot 11$ $265=5\cdot 53$

$266=2\cdot 7\cdot 19$ $267=3\cdot 89$ $268=2\cdot 2\cdot 67$ $269=269$

$270=2\cdot 3\cdot 3\cdot 3\cdot 5$ $271=271$ $272=2\cdot 2\cdot 2\cdot 2\cdot 17$ $273=3\cdot 7\cdot 13$

274=2·137 275=5·5·11 276=2·2·3·23 277=277
278=2·139 279=3·3·31 280=2·2·2·5·7 281=281
282=2·3·47 283=283 284=2·2·71 285=3·5·19
286=2·11·13 287=7·41 288=2·2·2·2·2·3·3 289=17·17
290=2·5·29 291=3·97 292=2·2·73 293=293
294=2·3·7·7 295=5·59 296=2·2·2·37 297=3·3·3·11
298=2·149 299=13·23 300=2·2·3·5·5 301=7·43
302=2·151 303=3·101 304=2·2·2·2·19 305=5·61
306=2·3·3·17 307=307 308=2·2·7·11 309=3·103
310=2·5·31 311=311 312=2·2·2·3·13 313=313
314=2·157 315=3·3·5·7 316=2·2·79 317=317
318=2·3·53 319=11·29 320=2·2·2·2·2·2·5 321=3·107
322=2·7·23 323=17·19 324=2·2·3·3·3·3 325=5·5·13
326=2·163 327=3·109 328=2·2·2·41 329=7·47
330=2·3·5·11 331=331 332=2·2·83 333=3·3·37
334=2·167 335=5·67 336=2·2·2·2·3·7 337=337
338=2·13·13 339=3·113 340=2·2·5·17 341=11·31
342=2·3·3·19 343=7·7·7 344=2·2·2·43 345=3·5·23
346=2·173 347=347 348=2·2·3·29 349=349
350=2·5·5·7 351=3·3·3·13 352=2·2·2·2·2·11 353=353
354=2·3·59 355=5·71 356=2·2·89 357=3·7·17
358=2·179 359=359 360=2·2·2·3·3·5 361=19·19
362=2·181 363=3·11·11 364=2·2·7·13 365=5·73
366=2·3·61 367=367 368=2·2·2·2·23 369=3·3·41
370=2·5·37 371=7·53 372=2·2·3·31 373=373
374=2·11·17 375=3·5·5·5 376=2·2·2·47 377=13·29
378=2·3·3·3·7 379=379 380=2·2·5·19 381=3·127
382=2·191 383=383 384=27·3 385=5·7·11
386=2·193 387=3·3·43 388=2·2·97 389=389
390=2·3·5·13 391=17·23 392=2·2·2·7·7 393=3·131
394=2·197 395=5·79 396=2·2·3·3·11 397=397
398=2·199 399=3·7·19 400=2·2·2·2·5·5 401=401
402=2·3·67 403=13·31 404=2·2·101 405=3·3·3·3·5
406=2·7·29 407=11·37 408=2·2·2·3·17 409=409
410=2·5·41 411=3·137 412=2·2·103 413=7·59
414=2·3·3·23 415=5·83 416=2·2·2·2·2·13 417=3·139
418=2·11·19 419=419 420=2·2·3·5·7 421=421
422=2·211 423=3·3·47 424=2·2·2·53 425=5·5·17
426=2·3·71 427=7·61 428=2·2·107 429=3·11·13

$430=2\cdot5\cdot43$ $431=431$ $432=2\cdot2\cdot2\cdot2\cdot3\cdot3\cdot3$ $433=433$

$434=2\cdot7\cdot31$ $435=3\cdot5\cdot29$ $436=2\cdot2\cdot109$ $437=19\cdot23$

$438=2\cdot3\cdot73$ $439=439$ $440=2\cdot2\cdot2\cdot5\cdot11$ $441=3\cdot3\cdot7\cdot7$

$442=2\cdot13\cdot17$ $443=443$ $444=2\cdot2\cdot3\cdot37$ $445=5\cdot89$

$446=2\cdot223$ $447=3\cdot149$ $448=2\cdot2\cdot2\cdot2\cdot2\cdot2\cdot7$ $449=449$

$450=2\cdot3\cdot3\cdot5\cdot5$ $451=11\cdot41$ $452=2\cdot2\cdot113$ $453=3\cdot151$

$454=2\cdot227$ $455=5\cdot7\cdot13$ $456=2\cdot2\cdot2\cdot3\cdot19$ $457=457$

$458=2\cdot229$ $459=3\cdot3\cdot3\cdot17$ $460=2\cdot2\cdot5\cdot23$ $461=461$

$462=2\cdot3\cdot7\cdot11$ $463=463$ $464=2\cdot2\cdot2\cdot2\cdot29$ $465=3\cdot5\cdot31$

$466=2\cdot233$ $467=467$ $468=2\cdot2\cdot3\cdot3\cdot13$ $469=7\cdot67$

$470=2\cdot5\cdot47$ $471=3\cdot157$ $472=2\cdot2\cdot2\cdot59$ $473=11\cdot43$

$474=2\cdot3\cdot79$ $475=5\cdot5\cdot19$ $476=2\cdot2\cdot7\cdot17$ $477=3\cdot3\cdot53$

$478=2\cdot239$ $479=479$ $480=2\cdot2\cdot2\cdot2\cdot2\cdot3\cdot5$ $481=13\cdot37$

$482=2\cdot241$ $483=3\cdot7\cdot23$ $484=2\cdot2\cdot11\cdot11$ $485=5\cdot97$

$486=2\cdot3\cdot3\cdot3\cdot3\cdot3$ $487=487$ $488=2\cdot2\cdot2\cdot61$ $489=3\cdot163$

$490=2\cdot5\cdot7\cdot7$ $491=491$ $492=2\cdot2\cdot3\cdot41$ $493=17\cdot29$

$494=2\cdot13\cdot19$ $495=3\cdot3\cdot5\cdot11$ $496=2\cdot2\cdot2\cdot2\cdot31$ $497=7\cdot71$

$498=2\cdot3\cdot83$ $499=499$ $500=2\cdot2\cdot5\cdot5\cdot5$ $501=3\cdot167$

$502=2\cdot251$ $503=503$ $504=2\cdot2\cdot2\cdot3\cdot3\cdot7$ $505=5\cdot101$

$506=2\cdot11\cdot23$ $507=3\cdot13\cdot13$ $508=2\cdot2\cdot127$ $509=509$

$510=2\cdot3\cdot5\cdot17$ $511=7\cdot73$ $512=2^9$ $513=3\cdot3\cdot3\cdot19$

$514=2\cdot257$ $515=5\cdot103$ $516=2\cdot2\cdot3\cdot43$ $517=11\cdot47$

$518=2\cdot7\cdot37$ $519=3\cdot173$ $520=2\cdot2\cdot2\cdot5\cdot13$ $521=521$

$522=2\cdot3\cdot3\cdot29$ $523=523$ $524=2\cdot2\cdot131$ $525=3\cdot5\cdot5\cdot7$

$526=2\cdot263$ $527=17\cdot31$ $528=2\cdot2\cdot2\cdot2\cdot3\cdot11$ $529=23\cdot23$

$530=2\cdot5\cdot53$ $531=3\cdot3\cdot59$ $532=2\cdot2\cdot7\cdot19$ $533=13\cdot41$

$534=2\cdot3\cdot89$ $535=5\cdot107$ $536=2\cdot2\cdot2\cdot67$ $537=3\cdot179$

$538=2\cdot269$ $539=7\cdot7\cdot11$ $540=2\cdot2\cdot3\cdot3\cdot3\cdot5$ $541=541$

$542=2\cdot271$ $543=3\cdot181$ $544=2\cdot2\cdot2\cdot2\cdot2\cdot17$ $545=5\cdot109$

$546=2\cdot3\cdot7\cdot13$ $547=547$ $548=2\cdot2\cdot137$ $549=3\cdot3\cdot61$

$550=2\cdot5\cdot5\cdot11$ $551=19\cdot29$ $552=2\cdot2\cdot2\cdot3\cdot23$ $553=7\cdot79$

$554=2\cdot277$ $555=3\cdot5\cdot37$ $556=2\cdot2\cdot139$ $557=557$

$558=2\cdot3\cdot3\cdot31$ $559=13\cdot43$ $560=2\cdot2\cdot2\cdot2\cdot5\cdot7$ $561=3\cdot11\cdot17$

$562=2\cdot281$ $563=563$ $564=2\cdot2\cdot3\cdot47$ $565=5\cdot113$

$566=2\cdot283$ $567=3\cdot3\cdot3\cdot3\cdot7$ $568=2\cdot2\cdot2\cdot71$ $569=569$

$570=2\cdot3\cdot5\cdot19$ $571=571$ $572=2\cdot2\cdot11\cdot13$ $573=3\cdot191$

$574=2\cdot7\cdot41$ $575=5\cdot5\cdot23$ $576=2^6\cdot3^2$ $577=577$

$578=2\cdot17\cdot17$ $579=3\cdot193$ $580=2\cdot2\cdot5\cdot29$ $581=7\cdot83$

$582=2\cdot3\cdot97$ $583=11\cdot53$ $584=2\cdot2\cdot2\cdot73$ $585=3\cdot3\cdot5\cdot13$

$586=2\cdot 293$	$587=587$	$588=2\cdot 2\cdot 3\cdot 7\cdot 7$	$589=19\cdot 31$
$590=2\cdot 5\cdot 59$	$591=3\cdot 197$	$592=2\cdot 2\cdot 2\cdot 2\cdot 37$	$593=593$
$594=2\cdot 3\cdot 3\cdot 3\cdot 11$	$595=5\cdot 7\cdot 17$	$596=2\cdot 2\cdot 149$	$597=3\cdot 199$
$598=2\cdot 13\cdot 23$	$599=599$	$600=2\cdot 2\cdot 2\cdot 3\cdot 5\cdot 5$	$601=601$

表 2　10 000 以内的素数表

3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97,
101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181,
191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277,
281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383,
389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487,
491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601,
607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709,
719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827,
829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947,
953, 967, 971, 977, 983, 991, 997, 1009, 1013, 1019, 1021, 1031, 1033, 1039, 1049,
1051, 1061, 1063, 1069, 1087, 1091, 1093, 1097, 1103, 1109, 1117, 1123, 1129, 1151,
1153, 1163, 1171, 1181, 1187, 1193, 1201, 1213, 1217, 1223, 1229, 1231, 1237, 1249,
1259, 1277, 1279, 1283, 1289, 1291, 1297, 1301, 1303, 1307, 1319, 1321, 1327, 1361,
1367, 1373, 1381, 1399, 1409, 1423, 1427, 1429, 1433, 1439, 1447, 1451, 1453, 1459,
1471, 1481, 1483, 1487, 1489, 1493, 1499, 1511, 1523, 1531, 1543, 1549, 1553, 1559,
1567, 1571, 1579, 1583, 1597, 1601, 1607, 1609, 1613, 1619, 1621, 1627, 1637, 1657,
1663, 1667, 1669, 1693, 1697, 1699, 1709, 1721, 1723, 1733, 1741, 1747, 1753, 1759,
1777, 1783, 1787, 1789, 1801, 1811, 1823, 1831, 1847, 1861, 1867, 1871, 1873, 1877,
1879, 1889, 1901, 1907, 1913, 1931, 1933, 1949, 1951, 1973, 1979, 1987, 1993, 1997,
1999, 2003, 2011, 2017, 2027, 2029, 2039, 2053, 2063, 2069, 2081, 2083, 2087, 2089,
2099, 2111, 2113, 2129, 2131, 2137, 2141, 2143, 2153, 2161, 2179, 2203, 2207, 2213,
2221, 2237, 2239, 2243, 2251, 2267, 2269, 2273, 2281, 2287, 2293, 2297, 2309, 2311,
2333, 2339, 2341, 2347, 2351, 2357, 2371, 2377, 2381, 2383, 2389, 2393, 2399, 2411,
2417, 2423, 2437, 2441, 2447, 2459, 2467, 2473, 2477, 2503, 2521, 2531, 2539, 2543,
2549, 2551, 2557, 2579, 2591, 2593, 2609, 2617, 2621, 2633, 2647, 2657, 2659, 2663,
2671, 2677, 2683, 2687, 2689, 2693, 2699, 2707, 2711, 2713, 2719, 2729, 2731, 2741,
2749, 2753, 2767, 2777, 2789, 2791, 2797, 2801, 2803, 2819, 2833, 2837, 2843, 2851,
2857, 2861, 2879, 2887, 2897, 2903, 2909, 2917, 2927, 2939, 2953, 2957, 2963, 2969,
2971, 2999, 3001, 3011, 3019, 3023, 3037, 3041, 3049, 3061, 3067, 3079, 3083, 3089,
3109, 3119, 3121, 3137, 3163, 3167, 3169, 3181, 3187, 3191, 3203, 3209, 3217, 3221,
3229, 3251, 3253, 3257, 3259, 3271, 3299, 3301, 3307, 3313, 3319, 3323, 3329, 3331,
3343, 3347, 3359, 3361, 3371, 3373, 3389, 3391, 3407, 3413, 3433, 3449, 3457, 3461,
3463, 3467, 3469, 3491, 3499, 3511, 3517, 3527, 3529, 3533, 3539, 3541, 3547, 3557,
3559, 3571, 3581, 3583, 3593, 3607, 3613, 3617, 3623, 3631, 3637, 3643, 3659, 3671,
3673, 3677, 3691, 3697, 3701, 3709, 3719, 3727, 3733, 3739, 3761, 3767, 3769, 3779,
3793, 3797, 3803, 3821, 3823, 3833, 3847, 3851, 3853, 3863, 3877, 3881, 3889, 3907,
3911, 3917, 3919, 3923, 3929, 3931, 3943, 3947, 3967, 3989, 4001, 4003, 4007, 4013,
4019, 4021, 4027, 4049, 4051, 4057, 4073, 4079, 4091, 4093, 4099, 4111, 4127, 4129,
4133, 4139, 4153, 4157, 4159, 4177, 4201, 4211, 4217, 4219, 4229, 4231, 4241, 4243,
4253, 4259, 4261, 4271, 4273, 4283, 4289, 4297, 4327, 4337, 4339, 4349, 4357, 4363,
4373, 4391, 4397, 4409, 4421, 4423, 4441, 4447, 4451, 4457, 4463, 4481, 4483,

4493, 4507, 4513, 4517, 4519, 4523, 4547, 4549, 4561, 4567, 4583, 4591, 4597, 4603, 4621, 4637, 4639, 4643, 4649, 4651, 4657, 4663, 4673, 4679, 4691, 4703, 4721, 4723, 4729, 4733, 4751, 4759, 4783, 4787, 4789, 4793, 4799, 4801, 4813, 4817, 4831, 4861, 4871, 4877, 4889, 4903, 4909, 4919, 4931, 4933, 4937, 4943, 4951, 4957, 4967, 4969, 4973, 4987, 4993, 4999, 5003, 5009, 5011, 5021, 5023, 5039, 5051, 5059, 5077, 5081, 5087, 5099, 5101, 5107, 5113, 5119, 5147, 5153, 5167, 5171, 5179, 5189, 5197, 5209, 5227, 5231, 5233, 5237, 5261, 5273, 5279, 5281, 5297, 5303, 5309, 5323, 5333, 5347, 5351, 5381, 5387, 5393, 5399, 5407, 5413, 5417, 5419, 5431, 5437, 5441, 5443, 5449, 5471, 5477, 5479, 5483, 5501, 5503, 5507, 5519, 5521, 5527, 5531, 5557, 5563, 5569, 5573, 5581, 5591, 5623, 5639, 5641, 5647, 5651, 5653, 5657, 5659, 5669, 5683, 5689, 5693, 5701, 5711, 5717, 5737, 5741, 5743, 5749, 5779, 5783, 5791, 5801, 5807, 5813, 5821, 5827, 5839, 5843, 5849, 5851, 5857, 5861, 5867, 5869, 5879, 5881, 5897, 5903, 5923, 5927, 5939, 5953, 5981, 5987, 6007, 6011, 6029, 6037, 6043, 6047, 6053, 6067, 6073, 6079, 6089, 6091, 6101, 6113, 6121, 6131, 6133, 6143, 6151, 6163, 6173, 6197, 6199, 6203, 6211, 6217, 6221, 6229, 6247, 6257, 6263, 6269, 6271, 6277, 6287, 6299, 6301, 6311, 6317, 6323, 6329, 6337, 6343, 6353, 6359, 6361, 6367, 6373, 6379, 6389, 6397, 6421, 6427, 6449, 6451, 6469, 6473, 6481, 6491, 6521, 6529, 6547, 6551, 6553, 6563, 6569, 6571, 6577, 6581, 6599, 6607, 6619, 6637, 6653, 6659, 6661, 6673, 6679, 6689, 6691, 6701, 6703, 6709, 6719, 6733, 6737, 6761, 6763, 6779, 6781, 6791, 6793, 6803, 6823, 6827, 6829, 6833, 6841, 6857, 6863, 6869, 6871, 6883, 6899, 6907, 6911, 6917, 6947, 6949, 6959, 6961, 6967, 6971, 6977, 6983, 6991, 6997, 7001, 7013, 7019, 7027, 7039, 7043, 7057, 7069, 7079, 7103, 7109, 7121, 7127, 7129, 7151, 7159, 7177, 7187, 7193, 7207, 7211, 7213, 7219, 7229, 7237, 7243, 7247, 7253, 7283, 7297, 7307, 7309, 7321, 7331, 7333, 7349, 7351, 7369, 7393, 7411, 7417, 7433, 7451, 7457, 7459, 7477, 7481, 7487, 7489, 7499, 7507, 7517, 7523, 7529, 7537, 7541, 7547, 7549, 7559, 7561, 7573, 7577, 7583, 7589, 7591, 7603, 7607, 7621, 7639, 7643, 7649, 7669, 7673, 7681, 7687, 7691, 7699, 7703, 7717, 7723, 7727, 7741, 7753, 7757, 7759, 7789, 7793, 7817, 7823, 7829, 7841, 7853, 7867, 7873, 7877, 7879, 7883, 7901, 7907, 7919, 7927, 7933, 7937, 7949, 7951, 7963, 7993, 8009, 8011, 8017, 8039, 8053, 8059, 8069, 8081, 8087, 8089, 8093, 8101, 8111, 8117, 8123, 8147, 8161, 8167, 8171, 8179, 8191, 8209, 8219, 8221, 8231, 8233, 8237, 8243, 8263, 8269, 8273, 8287, 8291, 8293, 8297, 8311, 8317, 8329, 8353, 8363, 8369, 8377, 8387, 8389, 8419, 8423, 8429, 8431, 8443, 8447, 8461, 8467, 8501, 8513, 8521, 8527, 8537, 8539, 8543, 8563, 8573, 8581, 8597, 8599, 8609, 8623, 8627, 8629, 8641, 8647, 8663, 8669, 8677, 8681, 8689, 8693, 8699, 8707, 8713, 8719, 8731, 8737, 8741, 8747, 8753, 8761, 8779, 8783, 8803, 8807, 8819, 8821, 8831, 8837, 8839, 8849, 8861, 8863, 8867, 8887, 8893, 8923, 8929, 8933, 8941, 8951, 8963, 8969, 8971, 8999, 9001, 9007, 9011, 9013, 9029, 9041, 9043, 9049, 9059, 9067, 9091, 9103, 9109, 9127, 9133, 9137, 9151, 9157, 9161, 9173, 9181, 9187, 9199, 9203, 9209, 9221, 9227, 9239, 9241, 9257, 9277, 9281, 9283, 9293, 9311, 9319, 9323, 9337, 9341, 9343, 9349, 9371, 9377, 9391, 9397, 9403, 9413, 9419, 9421, 9431, 9433, 9437, 9439, 9461, 9463, 9467, 9473, 9479, 9491, 9497, 9511, 9521, 9533, 9539, 9547, 9551, 9587, 9601, 9613, 9619, 9623, 9629, 9631, 9643, 9649, 9661, 9677, 9679, 9689, 9697, 9719, 9721, 9733, 9739, 9743, 9749, 9767, 9769, 9781, 9787, 9791, 9803, 9811, 9817, 9829, 9833, 9839, 9851, 9857, 9859, 9871, 9883, 9887, 9901, 9907, 9923, 9929, 9931, 9941, 9949, 9967, 9973.

表3 100以内的本原根

3的本原根为2。
5的本原根为2、3。
7的本原根为3、5。

11 的本原根为 2、6、7、8。
13 的本原根为 2、6、7、11。
17 的本原根为 3、5、6、7、10、11、12、14。
19 的本原根为 2、3、10、13、14、15。
23 的本原根为 5、7、10、11、14、15、17、19、20、21。
29 的本原根为 2、3、8、10、11、14、15、18、19、21、26、27。
31 的本原根为 3、11、12、13、17、21、22、24。
37 的本原根为 2、5、13、15、17、18、19、20、22、24、32、35。
41 的本原根为 6、7、11、12、13、15、17、19、22、24、26、28、29、30、34、35。
43 的本原根为 3、5、12、18、19、20、26、28、29、30、33、34。
47 的本原根为 5、10、11、13、15、19、20、22、23、26、29、30、31、33、35、38、39、40、41、43、44、45。
53 的本原根为 2、3、5、8、12、14、18、19、20、21、22、26、27、31、32、33、34、35、39、41、45、48、50、51。
59 的本原根为 2、6、8、10、11、13、14、18、23、24、30、31、32、33、34、37、38、39、40、42、43、44、47、50、52、54、55、56。
61 的本原根为 2、6、7、10、17、18、26、30、31、35、43、44、51、54、55、59。
67 的本原根为 2、7、11、12、13、18、20、28、31、32、34、41、44、46、48、50、51、57、61、63。
71 的本原根为 7、11、13、21、22、28、31、33、35、42、44、47、52、53、55、56、59、61、62、63、65、67、68、69。
73 的本原根为 5、11、13、14、15、20、26、28、29、31、33、34、39、40、42、44、45、47、53、58、59、60、62、68。
79 的本原根为 3、6、7、28、29、30、34、35、37、39、43、47、48、53、54、59、60、63、66、68、70、74、75、77。
83 的本原根为 2、5、6、8、13、14、15、18、19、20、22、24、32、34、35、39、42、43、45、46、47、50、52、53、54、55、56、57、58、60、62、66、67、71、72、73、74、76、79、80。
89 的本原根为 3、6、7、13、14、15、19、23、24、26、27、28、29、30、31、33、35、38、41、43、46、48、51、54、56、58、59、60、61、62、63、65、66、70、74、75、76、82、83、86。
97 的本原根为 5、7、10、13、14、15、17、21、23、26、29、37、38、39、40、41、56、57、58、59、60、68、71、74、76、80、82、83、84、87、90、92。